Intelligent Spatial Decision Support Systems

Raghbir Singh Sandhu

a thesis submitted for the degree of

Doctor of Philosophy in Computer Science

University of London

Department of Computer Science
University College London
Gower Street
London WC1E 6BT

June 1998

Abstract

This thesis investigates the conceptual and methodological issues for the development of **Intelligent Spatial Decision Support Systems** (**ISDSS**). These are spatial decision support systems (SDSS) integrating intelligent systems techniques (Genetic Algorithms, Neural Networks, Expert Systems, Fuzzy Logic and Nonlinear methods) with traditional modelling and statistical methods for the analysis of spatial problems.

The principal aim of this work is to verify the feasibility of heterogeneous systems for spatial decision support derived from a combination of traditional numerical techniques and intelligent techniques in order to provide superior performance and functionality to that achieved through the use of traditional methods alone.

This thesis is composed of four distinct sections: (i) a taxonomy covering the employment of intelligent systems techniques in specific applications of geographical information systems and SDSS; (ii) the development of a prototype ISDSS; (iii) application of the prototype ISDSS to modelling the spatiotemporal dynamics of high technology industry in the South-East of England; and (iv) the development of ISDSS architectures utilising interapplication communication techniques.

Existing approaches for implementing modelling tools within SDSS and GIS generally fall into one of two schemes - loose coupling or tight coupling – both of which involve a trade-off between generality and speed of data interchange. In addition, these schemes offer little use of distributed processing resources.

A prototype ISDSS was developed in collaboration with KPMG Peat Marwick's High Technology Practice as a general purpose spatiotemporal analysis tool with particular regard to modelling high technology industry. The ***GeoAnalyser*** system furnishes the user with animation and time plotting tools for observing spatiotemporal dynamics; such tools are typically not found in existing SDSS or GIS. Furthermore, *GeoAnalyser* employs the client/server model of distributed computing to link the front end client application with the back end modelling component contained within the server application. *GeoAnalyser* demonstrates a hybrid approach to spatial problem solving – the application utilises a nonlinear model for the temporal evolution of spatial variables and a genetic algorithm for calibrating the model in order to establish a good fit for the dataset under investigation.

Several novel architectures are proposed for ISDSS based on existing distributed systems technologies. These architectures are assessed in terms of user interface, data and functional integration. Implementation issues are also discussed.

The research contributions of this work are four-fold: (i) it lays the foundation for ISDSS as a distinct type of system for spatial decision support by examining the user interface, performance and methodological requirements of such systems; (ii) it explores a new approach for linking modelling techniques and SDSS; (iii) it investigates the possibility of modelling high technology industry; and (iv) it details novel architectures for ISDSS based on distributed systems.

Acknowledgements

This thesis was only possible with the assistance and advice of many people. I would first like to sincerely thank my supervisor Professor Philip Treleaven for his support, encouragement and guidance through all aspects of the research.

My gratitude is extended to Andrew Newland who, in his capacity as Manager of the High Technology Practice at KPMG Peat Marwick, provided very useful insights during the early development stages of the ***GeoAnalyser*** system.

I am also grateful to colleagues within the Computer Science department for their encouragement and advice during the completion of this work. Unfortunately there are far too many for me to name here but I would like in particular to thank Dr Sukhdev Khebbal for his inspiration and assistance while I was getting accustomed to UCL and throughout my time here. I also appreciate the assistance of Dr Jerry Severwright, Dr Nigel Pugh and Sheep T. Iconoclast of the Bartlett School of Architecture for giving up his time on many occasions to assist on issues of Macintosh programming.

I am indebted to my family for their continued patience. In particular I would like to thank my wife Sarabjit Kaur for her encouragement – she has been a great support to me since we were married and made me the happiest man in the world after giving birth to our daughter, Amman Kaur.

Finally, I would like to thank both KPMG Peat Marwick and the Engineering and Physical Sciences Research Council for their valuable financial support.

I know not knowledge, meditation and virtuous deeds, nor know Thy worth, O Lord. The Greatest of all is the True Guru Nanak, who has saved my honour in this age of darkness.

(Sri Guru Granth Sahib, p750)

Whatever work thou desirest to do, tell that to the Lord. He shall accomplish thy affair, the True Guru bears true testimony to it.

(Sri Guru Granth Sahib, p91)

Dedicated to my parents,

Gurdip Singh
and
Amrik Kaur

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter introduces the arguments and implementation issues for integrating intelligent systems techniques with spatial decision support systems. It also describes the motivation and goals, research contributions and organisation of this thesis.

1.1 Overview

Conventional systems for solving general spatial problems fall into one of two categories: Geographical Information Systems (GIS) and Spatial Decision Support Systems (SDSS). These systems provide various types of spatial analysis and visualisation tools contained within some form of problem-solving framework. This research proposes to augment the tools available in current systems by including *Intelligent Systems* techniques in the set of available tools, thereby forming *Intelligent Spatial Decision Support Systems (ISDSS).* These techniques offer a rich variety of problem solving features not available in standard mathematical and statistical methods.

1.2 Spatial Analysis and Problem Solving

Spatial analysis has been described as a set of techniques which operate on objects or events generating results that depend on both the location of the objects and their attributes (Goodchild 1987). The set of techniques covers descriptive measures of patterns of events as well as complex statistical analyses of events and processes. Application areas include land use planning, urban and regional modelling, and environmental modelling. The types of problems generally fall into the following categories:

- **Spatial query** – finding spatial features (points, lines and polygons) that match specified criteria. Queries are constructed using either text (often based on a form of the Structured Query Language, SQL) or selections (using a mouse) or a combination of the two.

- **Simulation** – using mathematical models of spatial processes to determine the effect of these processes on supplied data. Among the most complex simulations examine the temporal evolution of spatial attributes employing models constructed using differential equations.

- **Location selection/allocation** – finding one or more spatial objects that represent an optimum according to the constraints applied. This is distinguished from spatial query by the use of often quite complex models describing the spatial phenomenon of interest. For example, the use of shopping models to determinine the best site for locating a new retail centre. Such models take into account the effects of the transport network (including the calculation of trip functions), locational characteristics of the local residential population, location of competitors, commuter flows, etc (Goodchild and Noronha 1987).

In addition, spatial problems can be characterised as either ***structured***, ***ill-structured*** or a combination of structured and ill-structured components (Han, Kim et al. 1991). Structured problems are typified by the existence of well-defined methods of solution, operational procedures and decision rules that can be obtained from other similar problems. Examples of such problems include finding spatial features that match explicit criteria (e.g. using buffering). Ill-structured problems on the other hand have no such clearly defined methods and, indeed, their solutions do not have a correct solution, only those that can be measured by terms such as 'good', 'bad' or 'reasonable'.

A further complexity is that the data being analysed are subject to the usual constraints of accuracy and completeness associated with most real world data. Similarly there are often logistical problems of collecting data of sufficient resolution over a large study area, thereby increasing the chances of introducing further, possibly systematic, errors. This necessitates the provision of methods to deal with errors if warranted by the nature of the problem.

Spatial problems are distinguished by the additional complexity introduced by data that are not only distributed in a finite area but also have a finite spatial extent. These characteristics are summarised in Table 1.1. Space introduces an additional structuring that needs to be taken into account. For example, a city can be modelled at a simplistic level as a set of variables associated with a point in a region. However, a more complete description would define the city in terms of many regions, each representing a neighbourhood or borough, with variables of interest defined for each of these regions. The multilevel transport system would be modelled as overlapping networks, with each link in the network containing information such as carrying capacity, average traffic flow, etc.

Characteristic	Description	Example
Data availability	Difficulties of data collection and storage. Data is often of a sensitive or confidential nature.	UK employment data by SIC code at ward level
Large data requirements	Many problems (particularly where modelling is an integral part of the problem-solving procedure) need a great deal of supporting data.	Retail store location
Data collation and formats	Spatial data often originates from a number of sources. Lack of format standards means that data transformations are typically part of the problem-solving process.	Combining data in TIGER files with Arc/Info files
Spatial correlation	Spatial variables and attributes usually demonstrate a relationship based on their relative position.	Unemployment data
Temporal correlation	For spatiotemporal data, there is also a correlation in the temporal dimension.	Traffic flows
Noisy data	Much spatial data is obtained through surveys and sampling and are therefore subject to stochastic and possibly systematic errors.	Temperature variations
Finite spatial extent	Most spatial data are associated with regions rather than points.	Regional population distribution
Multilevel complexity	The data and solution spaces are typically very large for spatial problems.	Integrated transport and urban models

Table 1.1 – General characteristics of spatial problem-solving

Such descriptions of spatial entities become considerably more complex if time is included as a factor in the analysis. Spatial correlations of data become echoed by temporal correlations. These multidimensional representations require complex models and tools for both analysis and viewing the results of the analysis.

Spatial analysis has evolved from the theories of location of economic activity developed by Weber and others. Weber's seminal work (Weber 1929) was primarily concerned with the location of new economic functions in undeveloped or agricultural areas. This was subsequently refined into a study of market areas (Lösch 1954) and later generalised and stated formally as a mathematical theory (Isard 1956). However, much of this early work remained largely theoretical, lacking sufficient data for rigorous testing and further exploration of the theories. Additionally, the lack of suitable tools for storing data, performing analysis (typically involving thousands of calculations in the evaluation of

even simple spatial functions) and displaying results in a meaningful form limited the study of any available data.

It was not until the arrival of the computer age that many of these difficulties were gradually overcome. Uptake of computers in research organisations and government census and planning agencies permitted the collection and dissemination of large quantities of information. This has led to new developments in spatial analysis, such as the application of new types of statistical analysis, decision support techniques, etc.. Dedicated computer systems for the storage, manipulation, query and display of spatial data have been the main underpinning of these developments.

1.3 GIS and Spatial Decision Support Systems

Geographical Information Systems combine powerful relational database technology with flexible display and query tools to facilitate the visual exploration of spatial data. They have greatly eased the difficulties associated with spatial data gathering, manipulation and storage. The types of analysis tools offered vary widely across implementations, but generally include the following operations:

- **Map overlay** – the combination of many types of data at the same spatial scale.
- **Buffering** – the calculation of zones around points, lines and polygons to represent the spatial extent of effects originating from source.
- **Query** – the selection of spatial features based on given criteria.

For more sophisticated analysis (including modelling) several GIS implementations offer facilities to extend the functionality of the system using either a built-in macro programming language or via calls to external modules (both predefined and custom). In this way, customised GIS applications can be built to suit any particular task.

However, there are several areas where GIS functionality fails to meet the requirements for tackling complex problems. Many of these difficulties arise from the cartographic history of GIS, their original design for solving *structured* problems and their reliance on relational database technology. The four main difficulties are:

1. **Limited data model.** Current GIS data models are generally limited to static data and offer no facility for storing time variations and handling time series.
2. **Limited scope for interaction.** The conceptual model adopted by GIS is typically one where interaction during analysis procedures is kept at a

minimum. While this is sufficient for cartographic analysis, many spatial models require a great deal of interaction with the user (such as for entering model parameters and calibrating the spatial model) and embedding such user interaction within a GIS is problematic.

3. **Lack of decision support infrastructure.** There are generally no facilities to manage the problem-solving process (e.g. to compare the results of alternative methods or parameter sets). The burden is therefore placed on the user to keep track of this process during the exploration of complex problems.

4. **Lack of exploratory analysis tools.** As described above, the types of analysis available in GIS are confined to tools for structured problem-solving – cartographic manipulation, query and limited spatial statistical analysis. However, there are many useful techniques for performing exploratory analysis (e.g. genetic algorithms) which are not available in current GIS. Such exploratory methods offer new insights for tackling ill-structured problems. That there is a need for such methods is pointed to by the widespread awareness of the merit of such problem-solving approaches (Hopkins 1984; Burrough 1986; Burrough 1990; Openshaw 1994).

Some of these deficiencies (namely *1*, *2* and *3* above) are addressed by *Spatial Decision Support Systems* (Armstrong and Densham 1990), a recent development from GIS. These offer data models that are better suited to modelling tasks (Hopkins and Armstrong 1985; Armstrong and Densham 1990), expanded scope for interaction (although there are still issues for further work; see Densham 1994) and provide a supporting framework for the problem-solving process (Han, Kim et al. 1991). SDSS are therefore much better suited to solving ill-structured problems that often require many model runs or alterations to model parameters in order to search the usually large solution space. The main area where both existing SDSS and GIS lack functionality for handling ill-structured problems is in the provision of *exploratory* methods of searching the solution space and new types of modelling techniques. Examples of such techniques include genetic algorithms, rule induction, fuzzy logic, expert systems, complex systems analysis and neural networks. So-called *intelligent* systems techniques offer unique tools for extracting knowledge from spatial data, pattern recognition, knowledge encapsulation and handling both data imprecision and imprecise concepts.

1.4 Intelligent Systems Techniques

Intelligent Systems techniques have been used successfully for a wide class of problems, ranging from the forecasting of economic time series to providing decision support shells

for complex software systems. Recently there has also been interest in the application of these techniques in spatial analysis and spatial decision support.

The term "Intelligent Systems" is used to collectively describe different types of techniques which, as their main common factor, employ or create knowledge. Each of the techniques relies on quite diverse algorithms and software design. Interestingly, most of the techniques take their inspiration from nature.

Intelligent Systems techniques can be broadly classified into two types depending on their mode of operation. *Symbolic processing* techniques (expert systems, rule induction, fuzzy logic) essentially manipulate symbols using rules in order to arrive at an outcome. They are useful in areas where knowledge can be explicitly defined, for example in the form of production rules, and where the steps taken to arrive at a solution and the solution itself have equal importance. On the other hand, *adaptive processing* techniques (genetic algorithms, neural networks) employ algorithms to extract knowledge leading to a solution to a defined problem. They are useful in areas where an analytical solution is difficult to obtain (such as in pattern recognition) and also where the problem domain is constantly changing (e.g. financial trading).

The comparative strengths and weaknesses of these techniques are summarised in Table 1.2.

The techniques can be applied to a wider class of problems if they are combined in the form of *intelligent hybrid* systems, thereby building on the individual strengths of the techniques and overcoming their weaknesses.

Intelligent systems offer great potential in the analysis of spatial problems. This thesis supports recent arguments put forward in the field (see Openshaw 1995 for example) that these techniques offer significant benefits for the analysis of spatial data offering methods of discerning relationships, finding patterns, reasoning and taking into account the fuzziness inherent in real world spatial data.

1.5 Motivation and Thesis Goals

The main goal of this research has been to investigate the basic issues behind the integration of intelligent systems techniques with GIS and spatial decision support systems in the construction of so-called intelligent spatial decision support systems (ISDSS). The major motivations and goals have therefore been:

Technique	Inspiration	Strengths	Weaknesses
SYMBOLIC TECHNIQUES			
Expert Systems	Human decision making	Easy to manage large number of rules; provides explanation of reasoning; production systems are easy to use.	Difficult to capture rules. Copes badly with uncertainty and errors.
Rule Induction	No parallel	Automatically generates rules from sample data.	Derived rules may not make much sense, even though they seem to fit the data.
Fuzzy Logic	Human decision making	Captures uncertainty (i.e. fuzziness) in human reasoning. Systems are therefore robust.	Difficult to derive the rules. Fuzzy membership functions are also difficult to define.
ADAPTIVE TECHNIQUES			
Genetic Algorithms	Processes of evolution and natural selection	Can discover new knowledge for a given problem domain. Evolves near-optimal solutions.	Does not often find the **best** solution.
Neural Networks	Function of brain tissue	Can learn patterns from representative data. Can generalise to take into account incorrect or incomplete data. Good fault tolerance.	Cannot provide reasoning behind an outcome

Table 1.2 – Comparison of Intelligent Systems techniques

- To investigate the fundamental theoretical and practical issues regarding the employment of intelligent systems techniques in spatial data analysis.
- To develop a classification of the various existing uses of intelligent techniques in spatial analysis.
- To determine the advantages and disadvantages of integrating intelligent techniques with GIS and SDSS.
- To evaluate the viability of utilising distributed computing methods in the construction of such systems.
- To design and implement a prototype ISDSS applied to a complex spatial analysis task. The task chosen was to model the spatiotemporal dynamics of high technology industry in the South-East of England.
- To assess feasible architectures for the construction of ISDSS. The various measures employed to evaluate these architectures include: ease and speed of

information exchange; effectiveness of the user interface; degree of modularity and use of distributed computing resources.

1.6 Research Contributions

The main contributions of this work to the fields of GIS and spatial decision support are:

- **The detailed examination of user interface, performance, methodological and implementation characteristics of ISDSS**. Based on an assessment of the limitations of current SDSS, arguments are developed for the integration of intelligent systems techniques. The key advantages of ISDSS are identified and software engineering approaches for their development are presented. The analysis lays a foundation for discussing ISDSS as a distinct type of system for spatial decision support.

- **The development of a classification scheme for ISDSS**. The taxonomy enables meaningful comparisons to be made between examples of such systems. The scheme takes into account the specific use made of intelligent techniques within the construction of the overall system.

- **The demonstration of distributed computing techniques as a valid method for linking modelling techniques and SDSS.** These approaches overcome the main difficulties associated with loosely coupled, tightly coupled and fully integrated systems. Distributed methods also offer design autonomy, implementation autonomy and parallel execution on distributed hardware. Consequently, these methods have many advantages for managing the computational and operational requirements of ISDSS.

- **The design and implementation of a distributed ISDSS for the development of spatial interaction models**. The ISDSS provides several spatiotemporal visualisation tools, plug-in model capability and offers the use of genetic algorithms for the calibration of spatial models.

- **The development of a spatial interaction model for high technology industry in South East England.** This industry was chosen because of its interesting spatial and temporal dynamics. Model development was based on detailed data analysis of the available data using the demonstration ISDSS. The model was also calibrated using the ISDSS.

- **The design and analysis of novel distributed architectures for ISDSS.** These offer clear ISDSS development routes that go beyond the types of

spatial analysis systems currently available. Each architecture demonstrates the benefits of distributed computing techniques for real world systems.

1.7 Thesis Organisation

This thesis is organised into 9 main chapters:

Chapter Two – Spatial Decision Support Systems, defines Spatial Decision Support Systems, how they differ fundamentally from geographical information systems and discusses the facilities that SDSS provide for spatial decision making. It introduces Intelligent Systems techniques and presents the integration issues involved in combining these techniques with SDSS. It reviews software integration technologies and surveys the extent to which these technologies are supported by existing SDSS and GIS.

Chapter Three – Intelligent Spatial Decision Support Systems: Issues and Taxonomy, presents Intelligent Spatial Decision Support Systems as a development from SDSS, incorporating the benefits that Intelligent Systems techniques can bring to bear on ill and semi-structured spatial problems. A novel classification scheme for characterising the various ways intelligent systems techniques can extend the functionality of SDSS is presented. Specific applications of these techniques in spatial decision making are discussed by way of example. Implementation issues concerning mechanisms for integrating the technologies are introduced.

Chapter Four – The GeoAnalyser Intelligent Spatial Decision Support System, describes the design and implementation of the GeoAnalyser system. The design rationale, user interface and system components (covering the display tools, model controls, spatiotemporal database, genetic algorithm and the spatial model) are presented in detail. The method used to integrate the distributed system components is discussed in detail. Efficiency and data structuring issues are also examined.

Chapter Five – Application of GeoAnalyser to Industrial Modelling, describes the application of the GeoAnalyser system to modelling the evolution of high technology industry in the South-East of England. The modelling time frame covers the years from 1982 to 1992. From a definition of this industry, a detailed summary of modelling factors is provided, leading to the development of a nonlinear growth model. Next, the results of using the genetic algorithm to calibrate the model parameters are presented. The chapter concludes with an analysis of the prediction accuracy of the model.

Chapter Six – Implementation Issues of Intelligent Spatial Decision Support Systems, assesses the user interface and implementation issues arising from the

development and application of the GeoAnalyser prototype. The assessment provides the basis for developing a foundation for ISDSS.

Chapter Seven – Implementations of Intelligent Spatial Decision Support Systems, presents four specific architectures for developing ISDSS. Each of the architectures employs distributed computing techniques. In each case, the system construction is described in detail. The architectures are assessed in terms of the level of user interface integration, data integration and problem solving capability.

Chapter Eight – Assessment, assesses the research work described in this thesis. The ISDSS taxonomy, the development and application of the GeoAnalyser system, and the architectures for intelligent spatial decision support are assessed and compared against possible alternatives.

Chapter Nine – Conclusions and Future Work, summarises and presents the conclusions of this thesis. Further developments of the GeoAnalyser system, ISDSS architectures and ISDSS generally are discussed.

Chapter 2

Spatial Decision Support Systems

This chapter defines Spatial Decision Support Systems, how they differ fundamentally from Geographical Information Systems (GIS) and discusses the facilities that SDSS provide for spatial decision making. It introduces Intelligent Systems techniques and presents the integration issues involved in combining these techniques with SDSS. It reviews software integration technologies and surveys the extent to which these technologies are supported by existing SDSS and GIS.

2.1 Introduction

The last ten years or so have witnessed the emergence of Geographical Information Systems (GIS) as the mainstream application platform for performing spatial analysis. However, while GIS undoubtedly furnish the user with a powerful set of tools for viewing spatial data and performing map-based analysis, they only provide limited support for more advanced methods of spatial inquiry and exploration (Armstrong and Densham 1990; Burrough 1990).

Spatial Decision Support Systems transcend and, in some respects, complement the feature set provided by Geographical Information Systems. To provide an understanding of how this class of system differs from GIS, various definitions of SDSS are examined and the facilities they provide for spatial decision support are described. This is followed by an overview of *Intelligent Systems techniques* and an introduction to the implementation issues surrounding their integration with existing GIS and SDSS.

2.2 Definitions

In arriving at a satisfactory definition for SDSS, it is interesting to note how others view GIS. As an example, GIS have been defined as 'a decision support system involving the integration of spatially referenced data in a problem-solving environment' (Cowen 1988). The misconception that GIS are essentially decision support systems appears to be quite widespread. This is quite understandable given recent developments in the GIS field (Bracken and Webster 1989):

'Since most modern GIS now incorporate analytical models, support high-level query languages, and are typically used to provide answers to ill-structured spatial problems,

they can be properly thought of as geographical decision support systems. A commercial GIS suite may therefore be thought of as a spatial DSS building tool'.

These definitions cast a very wide net, covering SDSS and even systems which may not result in the creation of map images at all. Indeed, on the subject of equating GIS with any type of geographic data processing system, Cowen argues further that 'such vague definitions are doing great disservice by allowing the label of GIS to be applied to almost any software system that can display a map or map-like image on a computer screen'.

An alternative definition is 'a system of hardware, software, data, people, organisations, and institutional arrangements for collecting, storing and analysing, and disseminating information about areas of the Earth' (Nyerges 1993). This broad definition is in recognition of the ancillary requirements and processes in creating spatial databases and the deployment of GIS in organisations.

Clearly there are widely differing perspectives as to what constitutes a GIS. Nyerges acknowledges this by noting that 'a single definition of GIS is too limited to offer a full, functional description.'

To provide an alternative starting point for defining SDSS, a definition for decision support systems might therefore be more instructive. The following six characteristics comprise one definition that might be applied here (Geoffrion 1983):

1. 'They are used to tackle ill- or semi-structured problems – these occur when the problem, the decision-maker's objectives, or both, cannot be fully and coherently specified.

2. They are designed to be easy to use, the often very sophisticated computer technology is accessed through a user-friendly front end.

3. They are designed to enable the user to make full use of all the data and models that are available, so interfacing routines and data base management systems are important elements.

4. The user develops a solution procedure using the models as decision aids to generate a series of alternatives.

5. They are designed for flexibility of use and ease of adaptation to the evolving needs of the user.

6. They are developed interactively and recursively to provide a multiple-pass approach which contrasts with the more traditional serial approach – involving clearly defined phases through which the system progresses.'

These characteristics provide a useful defining framework for a wider class of problems than SDSS. A reasonable definition can be constructed by limiting this to problems involving spatial concepts and operating on spatial data in a non-trivial manner (the requirement of non-triviality excludes systems where the spatial nature of data plays no part in the solution of the problem).

Compare with the following definition which has been proposed (Armstrong and Densham 1990):

'A SDSS is focused on a limited problem domain, makes use of a variety of data types, brings analytical and statistical modelling capabilities to bear on problems, relies on graphic displays to convey information to decision-makers, is adaptable to the decision-maker's style of problem-solving and can easily be modified to include new capabilities.'

The ingredient missing from this view is support for the exploration of ill- or semi-structured problems and their solutions. This is provided by the first characteristic proposed by Geoffrion. The following amalgamates and summarises the above definitions:

'A system which is designed to support the solution of ill- or semi-structured spatial problems. The system flexibly and interactively aids the user in finding alternative solutions drawing on data of various types, including spatial data, using spatial analysis and modelling tools accessed via an intuitive user interface (incorporating graphic displays of output). It is adaptable to various problem-solving styles and typically can be adapted by the inclusion of new capabilities.'

This provides a useful set of ***defining properties*** which clearly anchors the conceptual nature of SDSS in decision support and the operational nature in spatial analysis.

Furthermore, as far as implementation is concerned, there is clearly much scope for developing both generic systems, capable of operating on a broad range of problems, as well as problem-specific SDSS. Given this broad scope of design goals, the following set of functional modules have been identified across SDSS implementations (Armstrong and Densham 1990):

1. Database management system;
2. Analysis routines;
3. Display and report generators; and
4. User interface.

As the authors note, this modular decomposition may or may not be reflected in the concrete design of the system as such. Rather, the decomposition is indicative of the different components that constitute a SDSS in the *functional* sense. The existence of such functional modules should be useful, in addition to the above definition, for deciding whether or not a particular system satisfies the criteria for a SDSS.

On a concluding note, it is interesting to point out that the definition of SDSS synthesised above contrasts sharply with widely accepted definitions of GIS (Burrough 1986; Goodchild and Kemp 1990; Star and Estes 1990; Bill and Fritsch 1991; Maguire 1991; Turk 1992; Fischer and Nijkamp 1993). However, as is the nature of classification, there will be systems that exhibit both GIS and SDSS properties. This does not pose a problem as far as the classification itself is concerned, because at this stage the definition is concerned mainly with the *properties* that are classed rather than system implementations. Therefore, the definition is more in the nature of a *conceptual framework* rather than a strict classification scheme.

2.3 Decision Support Approaches

In any given domain, problem solving involves the following basic procedure (Armstrong, De et al. 1990):

1. 'Determining what the real problem is,
2. Determining a strategy to find a solution, and
3. Finding a solution.'

For many classes of problem, all three of these steps are clearly defined and thus the solution procedure is straightforward. These are termed *well-defined* problems.

However, in those cases where 1 and 2 are not clearly defined, the process of finding a solution is not so straightforward. These are termed *ill-structured* or *semi-structured* problems. For many of these problems, determining the problem can be the most important and difficult part of arriving at a solution. For locational decision making problems, Armstrong et al point out that this lack of structure arises due to difficulty in:

1. 'Specifying a comprehensive set of relevant criteria at the outset...;
2. Specifying the weights to be assigned to criteria before decisionmakers know the characteristics of the system that best meets their weighted criteria;
3. Specifying site-specific constraints before decisionmakers know the consequence of imposing these constraints;

4. Capturing all relevant data about the geographical environment at a level of accuracy sufficient to discriminate between alternative solutions.'

To generalise, for any given spatial problem there can be great difficulties in being able to specify the problem in its entirety, which leads to obvious difficulties in solving such ill- or semi-structured problems. SDSS should permit the decision maker to evolve a better understanding of the problem and facilitate exploration of the problem domain using an extensible set of tools without imposing any particular style of problem-solving. In some cases, they actively assist the decision maker in the search for a problem definition and solution strategy.

It is worthwhile examining the type of decision support that SDSS implementations offer for spatial problem-solving. It has already been suggested that SDSS, being essentially a restricted subclass of DSS, provide some form of support for ill- or semi-structured spatial problems. What is the nature of that support? In what ways do they augment the decision making process so that real solutions may be found? To answer these questions, it would be instructive to examine some real examples of SDSS. Table 2.1 summarises the decision support features of various SDSS.

The systems described in Table 2.1 exhibit a combination of three fundamental decision support characteristics:

- ***Interactivity*** – A flexible user interface which facilitates the user to (possibly recursively) explore the problem in a number of prescribed ways, without locking the user into a fixed problem-solving procedure.

- ***Exploratory tools*** – A combination of tools for information visualisation and reporting, what-if analysis, spatial analysis, modelling and simulation. These tools assist in finding solutions to clearly defined sub problems. The emphasis should be on *exploration* rather than *solution*.

- ***Explicit decision support*** – Active assistance in the specification and solution of ill- or semi-structured problems. Tools include knowledge-based planners and decision support shells (allowing alternative solutions to be tracked and compared).

These characteristics serve to classify the type and nature of decision support provided by SDSS. The range of decision support features provided by a system is governed by a range of factors, such as cost, the original system design objectives, the nature of the human decision making process that is being facilitated, organisational processes, etc.

SDSS	Application	Decision Support Facilities	Other Features	Reference
MLDSS	Retail store location	Interactive, exploratory analysis	Uses a spatial interaction model with bicubic spline interpolation. 3D graphics are used to depict retails trade areas.	Kohsaka 1993
EDSS-1	Environmental management	Knowledge-based planning	Analytical models stored in a model base.	Guariso and Werthner 1989
KBDSS	Locational planning	Knowledge-based planning	Metaplanner (using the knowledge base).	Armstrong, De et al. 1990
DDSS	Distribution vehicle scheduling	Interactive analysis		Evans and Norback 1985
GADS	Locational planning	Exploratory analysis	DSS generator for a class of locational planning problems.	Carlson, Bennett et al. 1974
UDMS	Settlement planning	Interactive, exploratory analysis	Modular system design.	Robinson 1983
NYCFP	Locational planning	Exploratory analysis	Produces heuristic decision rules.	Blum 1972
SDSS for Reorganising Service Delivery Systems	Locational planning	Interactive, exploratory analysis allowing comparisons to be made between alternative solutions	Uses distributed computing resources for analysis and mapping.	Armstrong, Rushton et al. 1991
SDSS for Location Selection	Location selection	Interactive, exploratory analysis	Links a commercial GIS (TransCAD) with a location-allocation modelling package.	Willer 1990
BANKSIA	Environmental decision support	Interactive, exploratory analysis	Built using SISKIT, a DSS toolkit.	Abel, Yap et al. 1992
ADAPT	Land use planning	Keeps track of decisions and their underlying reasons.	Uses knowledge based methods to determine activity allocation.	Davis and Grant 1987
XPLANNER	Facility planning and management	Compares different plans; allows what-if simulations	Uses expert system for domain specific knowledge.	Han, Kim et al. 1991
ISIS	Streams management and planning	Interactive, exploratory analysis	Uses analytic database for decision support.	Hopkins and Armstrong 1985
GIMS	Land resource management	Interactive, exploratory analysis	Natural language query interface.	Kessel 1990

Table 2.1 – Decision support features of SDSS

Many of the systems in Table 2.1 have been developed using the traditional Operational Research methodology 'with the analyst stripping away the poorly structured decision environment to consider only the well-structured aspects of a problem' (Densham and Rushton 1988). Certainly, for most semi-structured problems, important parts of the problem are well-defined and can be solved using analytical and modelling methods. For such problems, there is great value in providing the analytical tools to help solve the structured components of the problem. These tools help the analyst effectively explore the solution space of a sub problem.

Another vital part of decision support is providing access to large quantities of relevant information, often stored in a database. The information might be transformed and depicted graphically (e.g. maps) as well as summarised in tables. Such tools play a valuable part in the decision process.

Some systems provide extensive support and guidance for problem specification, exploration of solutions and evaluation of outcomes. However, automated support of this nature is not required for certain decision processes, for example in complex applications where the decision environment is constantly changing such that the system cannot deal with all situations that are likely to arise. In these applications, the burden of keeping detailed records of potential solutions and their acceptance by the various parties involved must be assumed by the decision maker or analyst.

Of course, the wider reality is that the environment in which decision support systems are deployed involves participation amongst several parties which often leads quite naturally to multiple conflicting objectives. Conflict resolution is thus a main ingredient in the solution process, but one that is not explicitly supported in most systems for the reasons outlined above.

Alternative types of decision support that have been proposed for inclusion in SDSS include techniques such as Multi-Criteria Evaluation (Carver 1991). This allows direct comparisons to be made between solutions that are weighted differently by the various decision makers involved in a problem.

The next chapter introduces issues and arguments for the development of Intelligent Spatial Decision Support Systems. These build on SDSS by augmenting the decision support facilities and analytical tools with "Intelligent Systems techniques". ISDSS are defined and classified in Chapter 3. The next section serves as an introduction to Intelligent Systems techniques and their applications. Section 2.5 examines some of the integration issues surrounding their integration with both SDSS and GIS.

2.4 Intelligent Systems Techniques

The term "Intelligent Systems techniques" is used to collectively group widely differing sets of techniques which, as their main common factor, employ or create knowledge. The techniques are useful for performing generic tasks such as pattern recognition, decision support and forecasting. Each technique (covering Neural Networks, Genetic Algorithms, Rule Based Systems and Nonlinear Dynamics) relies on quite diverse algorithms and software construction. Most of the techniques take their inspiration from nature (e.g. neural networks are loosely based on the structure of brain tissue).

Intelligent Systems techniques have been utilised successfully for a wide class of problems, ranging from the forecasting of economic time series and customer profiling to providing decision support shells for complex software systems and optimising retail store layouts. More recently, their advantages over traditional statistical techniques have led to widespread application in the finance industry (Treleaven and Goonatilake 1992).

Intelligent Systems techniques offer unique advantages for many types of problem, including the following. These are summarised below:

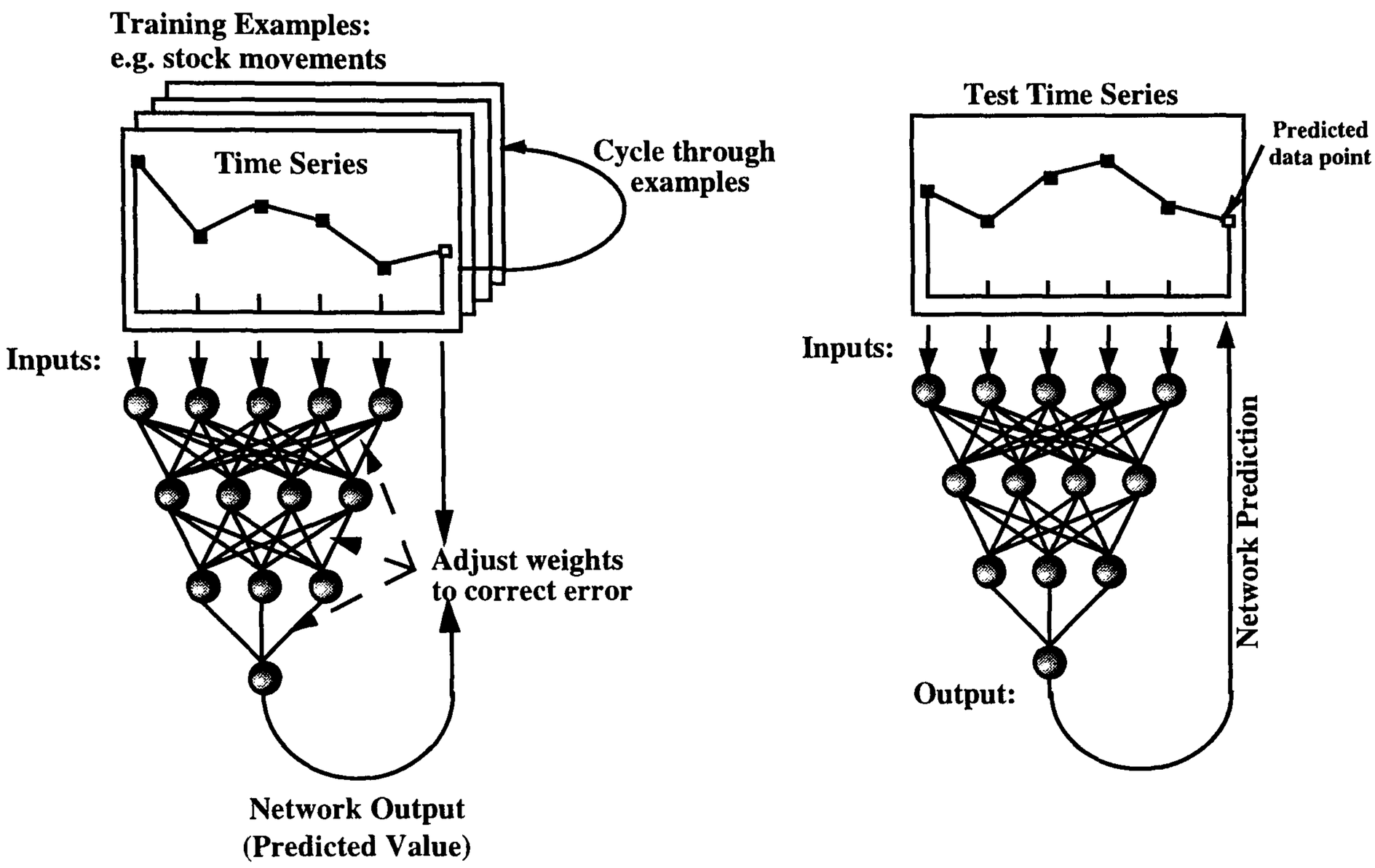
- **Flexibility** – the potential to be applied to a wide range of problems.
- **Adaptability** – slight changes in the problem domain can be accommodated automatically.
- **Learning** – past data can be analysed to solve a given problem without the need to define the individual steps in the solution.
- **Innovation** – the ability to discover new knowledge.

A brief introduction to the various techniques is provided below. The basic algorithms are described together with a summary of their main application areas, particular problem-solving strengths and applications in spatial problem-solving.

2.4.1 Neural Networks

As is well known, Neural Networks (Aleksander and Morton 1990) are distributed processing systems based loosely on the structure of the brain and the function of individual neurons. They are composed of a large number of simple processing units (*neurons*) linked by connections to form a network (as in Figure 2.1). Each of the connections has associated with it a variable weight which attenuates the strength of the signal transmitted by the connection. During execution, a neuron performs a simple sum of its input values (which may be coming from other neurons in the network). Next a

threshold is applied and the neuron only fires (i.e. transmits a signal at its output) if the sum of inputs exceeds that threshold. The process of adjusting the network weights is termed *learning* and is achieved by means of a *learning algorithm*. Two basic forms of learning exist. In *unsupervised learning*, a set of representative training examples are presented to the network, which classifies the data by making associations among them. In the second form, called *supervised learning*, both training examples and a desired network response is supplied. Here, the network learns to associate the arbitrary input patterns with the supplied output. The latter form of learning, being the more common, shall be demonstrated by means of the example below.



(a) Network training **(b) Prediction**

Figure 2.1 – Time-series prediction using a feed-forward Neural Network

A variety of network topologies exist. The most widespread is the *feed-forward* network using the *back-propagation* learning algorithm. This is briefly discussed below.

In a feed-forward network, the neurons are arranged in a number of layers, where each layer is fully connected to the neurons in the next layer. The middle layers are the so-called hidden layers, which are important as this is where feature representations are built up. This approach prohibits feedback loops which are very difficult to train. The algorithm proceeds by selecting a training example from the representative training dataset. On presentation to the network, appropriate input neurons fire, propagating a set of signals to the next layer, and so on. The signal which reaches the output is compared with the desired output (the next data point in the time series in this case) to calculate the network error. A share of this error is then propagated *backwards* from layer to layer

depending on the strength of the weights. The weights are then altered slightly (according to the *learning rate*) to compensate for the error. In this way it is hoped that the network will gradually arrive at a state where it has adequately learnt to associate any given input pattern. To test the network, unseen data is presented to the data and the network's prediction compared to the expected data.

A properly trained network can generalise from the examples in the training set and can deal with noisy or missing data. These strengths have led to successful applications in areas such as financial forecasting, customer profiling and pattern recognition.

In practice, it is actually quite difficult to decide the best values for the neuron threshold and learning rate. These difficulties can lead to *over-training*, where the network learns to treat each input pattern as a special case rather than classifying their common characteristics.

Neural networks are ideally suited for classification and pattern recognition tasks. No detailed knowledge about the data is required for the network to automatically define pattern classes which capture the essence of the input data. Unfortunately, neural networks cannot easily be endowed with the ability to explain the reasoning used to arrive at a decision. This limits the usefulness of this technique in certain areas (e.g. medical diagnosis).

Openshaw has demonstrated the usefulness of neural networks for modelling spatial interaction (Openshaw and Turton 1994) and classifying spatial data (Openshaw 1994, Openshaw and Wymer 1991). Wang has shown that neural networks can be successfully integrated with a GIS for performing land suitability analysis (Wang 1994).

2.4.2 Genetic Algorithms

Genetic Algorithms (Holland 1975, Davis 1991, Goldberg 1989) are increasingly popular probabilistic search methods inspired by the process of natural selection and evolution of the fittest. A brief overview of the key aspects of the technique are described below, making reference to the diagram in Figure 2.2.

The first stage in the construction of a Genetic Algorithm (GA) is *representation* - selecting the most useful method of describing the individual (i.e. a solution to the problem to be solved). Several approaches are possible, including the use of binary strings, alphanumeric types, integers and floating point numbers. Regardless of which form is adopted, individuals will be decomposed into genes and chromosomes (collections of genes). It is important that genes responsible for coding similar aspects of a problem (and are therefore linked in some way) are kept close together in the gene

sequence and, preferably, stored in a separate chromosome if effective solutions are to be found .

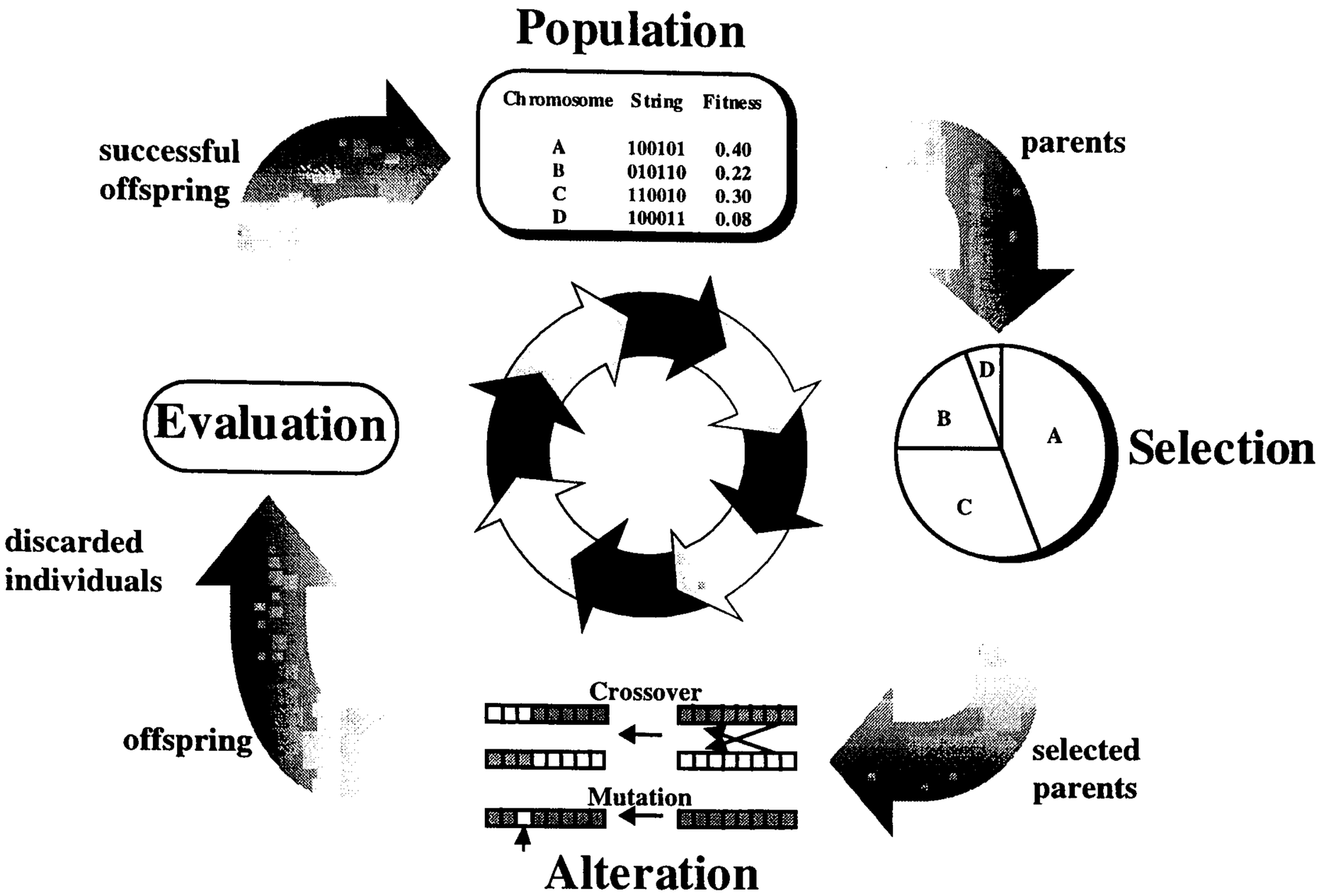


Figure 2.2 – Genetic Algorithm operation cycle

The GA commences with *Initialisation*. For this, one or more pools (i.e. collections) of individuals are created, with each individual's genes set to random values. Next the fitness of each individual is calculated using the *fitness function*. This function returns a value (normally a floating point number) reflecting the success of an individual at solving the problem at hand. The fitness values are used to perform *Selection* - determining which individuals are to be selected for reproduction. One common way of doing this is by Roulette Wheel selection (where the fitness values indicate the selection probability).

In the reproduction stage, *genetic operators* are employed to alter gene values and create offspring which differ from their parents. The *Crossover* operator is used to swap lengths of genes from corresponding chromosomes of individuals in the same pool. The *Mutation* operator introduces small changes in the gene pattern of an individual. This may cause a much larger change in an individual's phenotype (i.e. the expressed genes). Finally the *Migration* operator is used to copy or move individuals between pools if more than one pool was initially created.

Finally, the worst performing individuals are removed from the pools and the above process is iterated until a suitable solution has evolved.

GAs are capable of performing a nonlinear search of the entire search space, homing in on areas which yield good solutions. One of the key features is that successful traits are passed on to the next generation. The use of the fitness measure during the selection and alteration operations ensures that the fittest individuals are more likely to dominate those which are not so good at solving the problem. Due to their highly separable nature, GAs are particularly amenable to implementation on parallel hardware, and numerous toolkits are available for doing this (Ribeiro-Filho and Treleaven 1994) e.g. the PAPAGENA programming environment developed at UCL.

GAs have been successfully used for search/optimisation problems in a variety of areas e.g. the data mining of very large databases (Goldberg 1989), economic modelling (Stender 1994), etc.

Some exploratory work has been done on evolving spatial models (Openshaw, Charlton et al. 1987, Openshaw 1988) and calibrating particular models. There remains a great deal of promise for building spatial exploration and search algorithms using genetic algorithms embedded within GIS.

2.4.3 Rule-based Systems

Rule-based Systems are comprised of three distinct areas: *Expert Systems*, *Fuzzy Logic* and *Rule Induction*. Each of these is based on the concept of a *rule*, a logical assertion relating one or more conditions to a number of outcomes.

An *Expert System* (Jackson 1986) is a system that contains the knowledge of a human expert and can be consulted as if it were an expert. The basic idea is to store a particular area of knowledge in the form of a *rule base* (i.e. a related collection of rules). Rule creation is a lengthy, time consuming process, often performed by interviewing the human expert. Once the knowledge has been represented in this manner, it may be consulted via text-based query (as in an expert system shell). The *inference engine* selectively applies rules which match the input expression in order to arrive at a final outcome. One important aspect of expert systems is that the reasoning process may be examined at any stage. This is vital for application areas which require a certain degree of accountability; for example, the legal and medical professions. Expert systems are best suited for knowledge that is well understood, easily formulated into rules and fairly stable. The last point is important, as the costs of re-engineering a rule base can be high and the inability to respond to changing knowledge can result in systems which do not perform in their optimal capacity. Examples of expert systems applications include computer-based training systems, insurance claims processing and medical diagnostic systems.

Expert systems have been extensively assessed and used for spatial decision support and knowledge encapsulation (Burrough 1992). Expert systems shells are potentially useful as decision aids in all aspects of GIS and SDSS, from the collection (sampling) and calibration of data to the analysis of these data and the construction of queries. Such applications of expert systems within GIS have led to these systems being termed *Intelligent GIS* (Burrough 1992). Expert systems can aid modelling activities further by providing additional modelling support and assistance in the deployment of techniques which are difficult to master (neural networks and genetic algorithms in particular), resulting in the construction of *hybrid systems* where the user still has some measure of control.

Fuzzy Logic (Zadeh 1984) generalises on the Boolean valued logic of expert systems to permit an infinite range of values between `True` and `False` (binary `1` and `0`). This allows the use of imprecise terms and concepts familiar to humans (such as recent, quite low, much older, etc.) in the construction of rules for expert systems. Fuzzy values are determined by *membership functions*, which describe the degree of set membership of a particular variable. Fuzzy logic is better able to deal with the inherent inexact nature of the real world, where both inputs and outputs cannot easily be categorised into clear cut classes. Due to these strengths, fuzzy logic is being exploited in all manner of control systems (ranging from washing machines to automobiles) and for pattern recognition tasks, such as handwriting recognition and land use classification. It has also been used successfully to add fuzzy query capabilities to conventional database management systems and handle noisy data in financial trading systems.

There has recently been a great deal of interest in using fuzzy methods to capture the inexactness in spatial data and define fuzzy terms to relate spatial entities (Altman 1994). Examples include using fuzzy algorithms for clustering spatial data (McBratney and Moore 1985), efficient methods of performing segment intersections during map overlay operations (Guevara and Bishop 1985), describing the gradual transition of spatial values (Sui 1992), defining spatial linguistic terms (Leung 1982) and the development of models for fuzzy relational databases supporting fuzzy queries (Robinson 1988; Wang 1994).

Finally *Rule Induction* (Forsyth 1986) is an important technique for automatically discovering rules from a sample data set. This is achieved by classifying the available data using statistical techniques in order to find relationships. One example is ID3 (Quinlan 1979), which relies on entropy measures. The resulting knowledge is stored as a *decision tree* and may be usefully incorporated into an expert system. Rule Induction is also of benefit for data mining, an increasingly important set of methods for gleaning significant information from large databases.

The ID3 rule induction algorithm has been used as part of XPLANNER (Han, Kim et al. 1991), a SDSS for facility management and planning, to generate production rules from sample decision data.

Each of the above techniques are complementary - both rule induction and fuzzy logic may be utilised to enhance expert systems - allowing the construction of superior hybrid knowledge systems (data permitting).

2.4.4 Nonlinear Dynamics

Nonlinear Dynamics (Thompson and Stewart 1986), also known as Dynamical Systems Theory or simply Chaos Theory, has emerged as a set of techniques for analysing natural processes which appear to be random but in fact have an underlying deterministic nature. Examples of such processes include fluid flow and the behaviour of financial markets (Peters 1991). The data from these systems, when plotted using suitable co-ordinates, can reveal a certain degree of structure in the dynamics. This structure is termed the *attracting set* (or *attractor*). The techniques embrace:

- **Characterisation** - quantifying the degree of nonlinearity and determinism via measures such as the Lyapunov exponent, fractal and Haussdorf dimension, etc.;

- **Embedding** - methods of observing the attractor via Poincaré sections and singular systems analysis;

- **Local and global mappings** - to predict future behaviour in the short and medium term (long term predictions are essentially impossible due to the sensitive dependence on initial conditions - small, immeasurable changes in the initial state of the system can cause a large change in the observed evolution).

Application areas which have derived benefit from these techniques include local and global weather forecasting, analysing engineering systems and predicting financial time series. Although nonlinear dynamics is not, strictly speaking, an Intelligent Systems technique, it has been included here because of the tools that it provides for dealing with complex nonlinear behaviour.

Nonlinear evolutionary spatial interaction models were quite popular during the 80's for modelling industrial growth (see for example Allen and Sanglier 1981). Aside from this, the lack of sufficient spatial time series data has prevented the application of these powerful methods of data analysis.

2.4.5 Hybrid Systems

Combining the best features of different methods is essentially the approach taken to construct Hybrid Systems. These often generate superior results to those obtained through the use of any particular technique. Various approaches to devising hybrid systems are possible. These generally fall into one of the following classes (Goonatilake and Khebbal 1995):

- **Function-replacing hybrids** - where part of the process of a single technique is replaced by another technique (e.g. using a genetic algorithm to optimise the weights in a neural network);

- **Intercommunicating hybrids** - here the problem is tackled by self contained techniques which exchange data (e.g. using a neural network to first classify inputs before being passed to an expert system);

- **Polymorphic hybrids** - an existing technique is simply used to replace the functionality of another (e.g. performing expert system-like symbolic reasoning with a neural network).

Further examples include employing an expert system to assist in the tricky process of configuring and training a neural network. Or a genetic algorithm could be utilised to evolve a set of rules for an expert system.

Some existing and potential application areas of intelligent techniques in spatial problem-solving are summarised in Figure 2.3.

Neural Networks	**Genetic Algorithms**	**Expert Systems**
• Seismic pattern recognition • Land use classification • Spatial pattern recognition • Spatial time series forecasting	• Regional modelling • Spatial location search • Facilities optimisation • Utilities distribution	• Decision support • Land use classification
Fuzzy Logic	**Rule Induction**	**Nonlinear Dynamics**
• Fuzzy spatial query and fuzzy databases • Decision support • Land use classification	• Spatial data mining • Transport modelling	• Climate change • Population dynamics • Spatial time series forecasting • Terrain characterisation

Figure 2.3 – Application of intelligent techniques for spatial problem-solving

2.5 Methods of Integration

Intelligent Systems techniques offer a wide range of powerful tools for exploring data of all types, including spatial data. There are three main software engineering methods that may be taken for the utilisation of these techniques in the construction of SDSS. First, *bespoke systems* can be constructed from scratch by building all the required tools into the one software system. This results in monolithic application programs, long product development times, inflexibility for future developments and, consequently, high costs. KBDSS (Armstrong, De et al. 1990) is an example of this approach.

Secondly, the system can be constructed in a *highly modular* fashion as a number of distinct software systems, with each module custom-built but operating reasonably independently from the others. This approach is still fairly costly, as each module uses a proprietary interface and is designed to work within the context of larger system, but has the advantage of offering the developer a more manageable development cycle. BANKSIA (Abel, Yap et al. 1992) has been built using this type of approach.

Thirdly, the system can be devised as a set of *interacting modules*, as with the second approach, but *using off-the-shelf software* wherever possible as well as custom-built systems. This has the advantage of having low development costs, flexible development cycle and ease of adding new features. It also allows the developer to create a feature-rich decision support system without requiring many person-years of effort to do so. Willer's SDSS for Location Selection (Willer 1990) is an example of this approach, linking a standard GIS (TransCAD) with a custom-built location-allocation modelling package.

There are considerable advantages in employing the third approach for developing decision support systems. The main hurdle in using this approach is in being able to transparently link together the various software subsystems so that they behave as a single system. Chapter 3 deals with this issue in more detail and proposes an alternative method of integrating different software subsystems. The *Client-Server Model*, widely used in industry as a method of software integration, is introduced in the following sections .

2.5.1 Client-Server Model

Over the past fifteen years or so, personal and office computing systems have evolved from standalone machines to computers linked by a communications network. This was driven chiefly by the greater flexibility afforded by being able to share valuable system resources, such as file stores, printers, etc. An essential requirement for achieving this shift towards decentralised facilities was a method of passing messages between applications across such a network.

In the software domain, the object oriented methodology has become another enabling force by allowing applications to be built from small, self-contained and reusable components. This has led in turn to far greater flexibility, adaptability and scalability in the software development process than was possible before. Distributed message passing facilities have also enabled the construction of systems that are object oriented at the *system* level. Examples include client-server databases, which are accessible objects in the system environment.

Utilisation of this technology has not, however, been mirrored in the GIS and SDSS field. The technology has received very little attention in the literature despite the general shift towards this type of system in the commercial domain. Nevertheless, this state of affairs has not prevented a few vendors from providing limited support for client-server technology in their products (see below).

Several classes of interapplication communication (IAC) technology are available. The technology provides the underlying basis for developing client-server applications that can co-operate across a network.

In the basic client-server model (Berson 1992), a *client* application requests services from a *server* application (see Figure 2.4). An example of such a service might be a request to send some data in an agreed format. The request would be packaged according to the IAC protocol supported and delivered by the operating system. The server application would then verify the request and send back either the data (if the request meets the appropriate criteria) or else generate an error message. The operating system ensures that the messages are delivered successfully and the applications concerned are responsible for dispatching and handling requests for services in the appropriate manner.

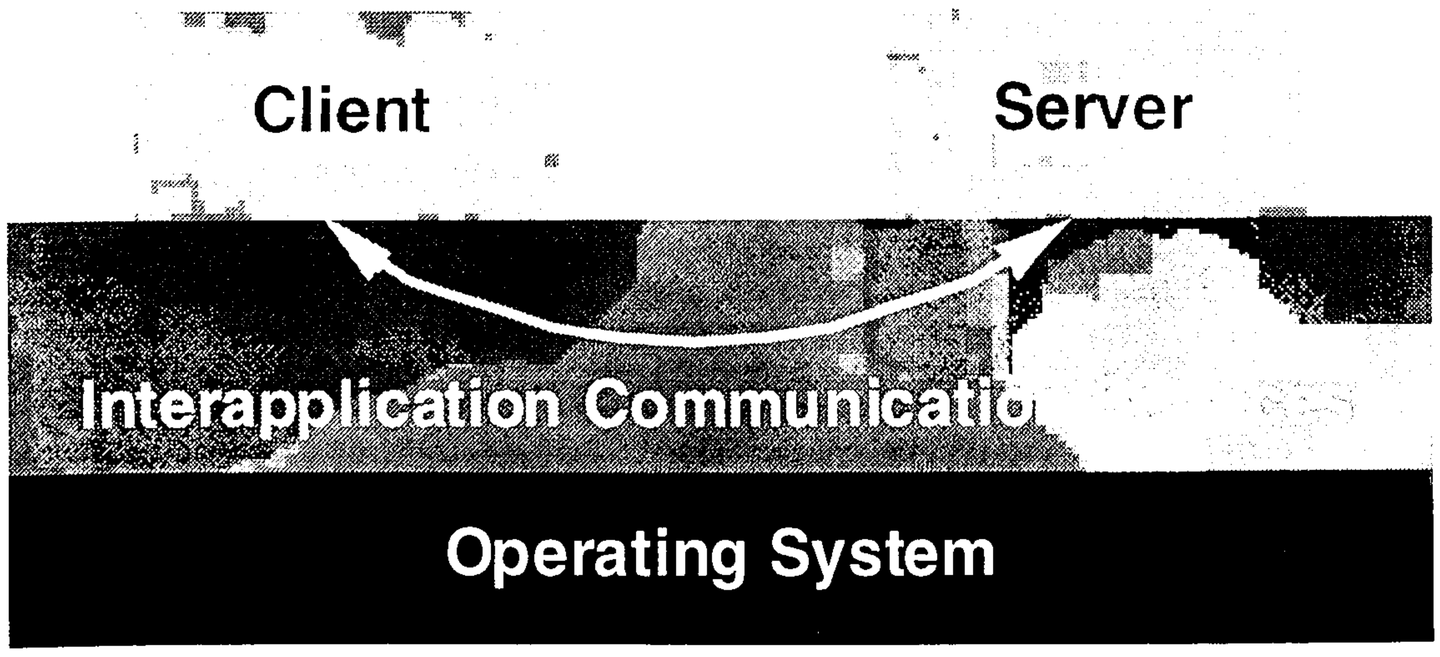


Figure 2.4 – Simple client-server model

Client-server systems operate either over a network or on a standalone machine. The advantage of running over a network is that they can make better use of hardware resources. For example, a fast computer with a large quantity of disk storage is a typical

requirement for a database server. Authorised users on the same network as the server can make use of this hardware, resulting in a more cost-effective set-up.

Another key feature of messaging passing techniques is that they permit applications to be built from smaller components that communicate with one another in a predefined way. This enables higher levels of code reuse, as many components can be built fairly independently from the main system. Modular or object oriented methods also make system design considerably easier, both by allowing the overall system to be conceptually broken down into smaller subsystems and by promoting clearer divisions of work between programming teams, as implementation dependencies are reduced.

The following sections discuss message passing facilities on three distinct operating systems: Microsoft Windows, Unix and Apple MacOS. These are essentially very similar technologies but they are inherently incompatible with one another and offer different network services.

2.5.2 Microsoft DDE

The Dynamic Data Exchange (Clark 1992) mechanism of Microsoft Windows implements an operating system-supported protocol for performing message passing between applications. DDE is also available in a similar form on IBM's OS/2 operating system.

A client typically engages in the following procedure to conduct a DDE message exchange (Vose 1990):

- *Allocate shared memory for a DDE object*
  Both applications will access this memory to exchange the DDE message.

- *Create a format for the data to be passed*
  This might be the supplied clipboard format, or any other custom format.

- *Select a type for the information exchange*
  This specifies whether the exchange will be once only, a warm link or a hot link.

- *Send the DDE message*

- *Deallocate shared memory*

The second step is important to note for our discussion. By specifying a special format for the data to be exchanged the structure of the data item is preserved. There is no need to export the data in, say, text format. This helps speed things up considerably provided both applications agree on the format. Note also that the exchange uses main memory - there is

no recourse to disk storage - resulting in a high operating speed. The third step permits dynamic data to be transferred - if the data on the server side is constantly changing then setting up a hot link will enable the client using that data to take account of those changes automatically. This gets around the update problem associated with data which have been statically cut and pasted into a document.

While DDE is primarily geared towards seamlessly integrating applications running on the same PC, it can also be used to link applications across a network. Using what are termed *redirectors*, the client initiates a message exchange with a local redirector. This then packages the information and sends it over the network to another redirector on the server machine, which decodes the information back into a DDE message and transfers it to the server application for processing.

Several GIS currently support the use of DDE to various degrees, including Tydac's SPANS MAP, Pafec GIS for Windows and MapInfo.

2.5.3 Unix Sockets and Pipes

Unix operating systems generally provide at least two methods of communicating amongst applications. Although there are higher level services available from software vendors, this discussion is limited to those available at the operating system level.

A *pipe* is a communication channel initiated by a running process (i.e. application) which behaves very much like standard Unix file redirection. A new process is created in parallel to the existing process and data can be piped to and from it as a sequence of bytes. The restriction here is that data can only be piped to a new process that resides on the same machine as the original process.

The second method, called *sockets*, relies on TCP/IP, the communications protocol underpinning Unix networks and the Internet (Comer and Stevens 1994). A socket is a one-way connection between two processes. Since each socket has associated with it an Internet machine address and port number, the client and server processes can reside on separate machines anywhere on the Internet. Communication is achieved by sending data via a local socket. This is then transmitted over the network to the remote socket from where it may be read by the server process. Typical client-server exchanges involve setting up two sockets at each end, one for sending data and the other for receiving.

Applications written using sockets are fairly portable due to the wide availability of this technology (implementations are even available for the other two operating systems discussed here). However, there are currently no GIS or SDSS implementations that provide an open interface for integrating using sockets.

2.5.4 Apple Events

The MacOS provides a number of related facilities for interapplication communication (see Apple Computer 1993 for further details).

At the lowest level, the *Program-to-Program Communications (PPC) Toolbox* provides an efficient mechanism for sending blocks of data between applications that may be spread across a network. This is mainly used by the operating system and applications that need to exchange large quantities of data.

The main IAC facility supported by MacOS is *Apple Events*, which is based on the services of the PPC Toolbox. All events, covering both user events (e.g. mouse clicks and key presses) and system events (e.g. opening an application), are Apple Events. Applications that support Apple Events must be capable of responding to a minimum of four required events and any number of custom events as well as other optional system events.

A preliminary step which must be negotiated by a client application is that of locating the server application. Since the server may also be located on another computer, a standard browser is provided to enable the user to specify which appropriate server application he or she would like to communicate with. The process for sending an Apple Event then proceeds as follows:

- The client creates an Apple Event object.
- Data items are packaged into the event object.
- The event object is then sent to the specified server.
- Any data returned from the server are retrieved before disposing of the event object.

An interesting feature is the ability to insert *object specifier records* within an event object. These provide a standard way of referring to particular elements of a document, such as "word 4 on line 2 of paragraph 3 of the document called Letter" or "region labelled London within document UK_populations". If the event is considered to be a verb (e.g. "Copy") the specifier records determine which object that verb should operate on.

Publish and Subscribe uses Apple Events to provide a user-oriented method of sharing dynamic information among a number of applications. From within a drawing package, for example, a user might publish an *edition* of a drawing. Then, from a word-processor,

this edition can be subscribed to and the drawing placed within the document. Any consequent changes to the drawing will be automatically reflected inside the text document. This is similar to hot-linking data using DDE but instead relies on files rather than shared memory to perform the exchange.

Custom and system Apple Events can be recorded as a *script*. The script can be executed any number of times to automate simple repetitive tasks. Scripts may also be compiled by hand and this provides a very powerful capability for assembling custom systems that use one or more component applications. The following is an example of such a script for selecting a specific region on a map:

```
tell application "MapMaker"

      select the regions of map MyMap

            whose attribute "population" >= 10000

end tell
```

Scripts can also be used to implement parts of an application's functionality, such as code for responding to a mouse click. It is then possible for the user to replace these scripts with their own, thereby modifying the standard behaviour of an application to suit a particular purpose.

The user is not restricted to using the standard Apple Script language. Through the *Open Scripting Architecture* of MacOS any number of custom scripting languages can be implemented, using language constructs which are appropriate to a particular application domain.

Currently, MapInfo is the only GIS on the Macintosh that supports the use of Apple Events (handled through the Map Basic macro language).

2.5.5 Summary of Integration Methods

The variety and type of IAC facilities on the platforms discussed indicates the lack of standards across the industry. Nevertheless a number of GIS vendors have built client-server interfaces into their systems. This presents the possibility of building SDSS by linking analytical tools with existing GIS using client-server methods. The technology provides a number of support services in addition to message passing protocols; some of these, notably the ability to refer to subelements of a document entity in a standard manner, could be very useful for interacting with spatial data sets accessed via a GIS.

2.6 Summary

This chapter has introduced SDSS as systems for dealing with ill- or semi-structured spatial problems. A clear distinction has been made between SDSS and GIS, which are essentially spatial data management, visualisation and analysis packages.

The types of decision support facilities that are built into existing SDSS have been surveyed. These range from providing a highly interactive user environment to including exploratory analysis and visualisation tools and endowing active support for the specification and solution of spatial problems.

A description of Intelligent Systems techniques has been given together with an account of their unique problem-solving and analytical functionality. Finally, implementation issues surrounding the integration of various software subsystems into a single SDSS were introduced. The client-server model and the underlying interapplication communication technologies have been described for a number of mainstream operating systems. These provide effective methods of integrating system components and utilising distributed hardware resources.

Chapter 3

Intelligent Spatial Decision Support Systems: Issues and Taxonomy

This chapter presents Intelligent Spatial Decision Support Systems as a development from SDSS, incorporating the benefits that Intelligent Systems techniques can bring to bear on ill and semi-structured spatial problems. Arguments are developed for the construction of these systems. The issues involved in building ISDSS are discussed. Existing methods of integrating analytical tools within GIS and SDSS are evaluated. Distributed computing methods are then proposed to provide a mechanism for integrating the software subsystems within ISDSS. Next, a classification scheme is presented for characterising the various ways in which Intelligent Systems techniques may be employed within ISDSS. To illustrate this taxonomy, specific applications of these techniques in spatial decision making are discussed.

3.1 Introduction

Intelligent Systems techniques offer many advantages over traditional methods (such as statistical analysis), particularly where the problem domain exhibits some form of nonlinearity. They have found applications in many fields, including those relating to spatial analysis, SDSS and GIS. This chapter describes a defining and conceptual framework for combining Intelligent Systems techniques with SDSS to form Intelligent Spatial Decision Support Systems (ISDSS). Starting with a formal definition, arguments are presented to highlight the need for ISDSS. The various issues associated with developing a new class of decision support system are discussed in depth. Following a discussion of coupling techniques currently employed, new alternatives based on the client-server model are offered to address the general issue of linking analytical software modules within SDSS and GIS.

3.2 Definition

At a simplistic level, ISDSS can be considered to be spatial decision support systems that employ Intelligent Systems techniques for the exploration of spatial problems. The benefits of utilising Intelligent Systems for spatial analysis have already been identified. However, their use need not be restricted to purely analytical functions. For example, KBDSS (Armstrong, De et al. 1990) employs expert systems to encapsulate and apply

domain specific knowledge for location selection of sites. They have also been successfully applied for data query, feature extraction, classification and spatial analysis. Section 3 details numerous examples of the use of Intelligent Systems for spatial analysis and decision support as part of the description of the ISDSS taxonomy.

The following definition, a modified form of the SDSS definition presented in Chapter 2, is proposed as a definition of Intelligent Spatial Decision Support Systems:

'A system which is designed to support the solution of ill- or semi-structured spatial problems. The system flexibly and interactively aids the user in finding alternative solutions drawing on data of various types, including spatial data, using Intelligent Systems techniques, traditional spatial analysis functions and modelling tools accessed via an intuitive user interface (incorporating graphic displays of output). It is adaptable to various problem-solving styles and is typically extensible by the inclusion of new analytical capabilities.'

Systems that exhibit these properties are considered to be a distinct class of SDSS. There are various reasons for defining ISDSS to be a subclass of SDSS. First, the conceptual design issues of ISDSS are more finely focused on knowledge processing in addition to decision support and analysis. Secondly, the definition establishes a distinct conceptual framework for combining novel computing techniques and existing methods to build new systems. Thirdly, the distinction aids clarification of the important issues in the application of Intelligent Systems techniques in spatial decision support.

3.3 Arguments for ISDSS

The benefits of using Intelligent Systems techniques for spatial decision support have been touched on briefly in earlier sections. The main points are described fully here with reference to arguments advanced in recent literature. These arguments also concern GIS in addition to SDSS because, as detailed in Section 3.4, GIS play a key role in many SDSS for delivering extensive visualisation and spatial data management facilities.

The following spatial decision support requirements are not being met adequately by existing systems:

- *Data saturation* – Improved data collection and storage methods has led to an explosion in the availability of all types of spatial data. However there is a paucity of methods for extracting useful knowledge from large datasets in existing systems (Openshaw 1991).

- *Data complexity* – Coupled with the explosion in data availability, it is clear that spatial data exhibits tremendous complexity in both the spatial and

temporal domains. The classical theories cannot cope with this type of complexity: "Sadly, we have not got very far . . . What exists is typically old (von Thünen, Christaller, Hoyt, etc.)" (Openshaw 1994). Often, the complex detail is treated as noise.

- *Limited modelling capabilities* – Existing modelling techniques offer a limited range of functionality for the analysis of spatial data. Specifically, they are geared for handling hard analysis problems and cannot deal with soft information. Furthermore, they offer limited scope for the exploratory analysis of large data sets (Openshaw 1994). Additionally, the analytical capabilities of current GIS are based on *cartographic* technology (Openshaw, Cross et al. 1990).

- *Model complexity* – Some of the available analysis tools are difficult to use: "It is not easy for users to learn how and when to use the spatial analysis tools to achieve any desired result" (Burrough 1992). Users of such tools need additional support in the form of expert guidance in order to use them effectively.

- *Lack of spatial decision support* – Few systems provide active support for decision making and solving complex ill-structured spatial problem. Expert knowledge is required for being able to specify and solve such problems (see for example Han, Kim et al. 1991)

These shortcomings are addressed to various extents by employing Intelligent Systems techniques. They enhance traditional techniques when used in conjunction with existing methods and they also offer completely new functionality (e.g. automated model generation Openshaw 1988). Three main areas benefit from combining these techniques with SDSS:

- *Enhancement of spatial modelling and spatial data analysis*. Intelligent Systems techniques significantly widen the set of analytical tools available (Smith 1984). Techniques such as neural networks and genetic algorithms can be used for mining and modelling large datasets (Openshaw 1994) and working with complex nonlinear spatial data. Fuzzy logic provides a better and more natural way of representing spatial relationships (Burrough 1989, Burrough 1986). Other examples include using rule based systems for land suitability assessment (Diamond and Wright 1988) and planning (Davis, Compagnoni et al. 1987).

- *Knowledge processing*. Rule based systems and neural networks are useful for eliciting, representing and applying soft domain knowledge.

- *Enhanced decision support*. Intelligent techniques, particularly expert systems, can assist in many stages of the decision process (Densham and Rushton 1988, Davis and Grant 1987): specifying the problem, selecting data, assistance with using models (Han, Kim et al. 1991, Burrough 1992), evaluating and comparing solutions, assessing reliability of results and treatment of errors.

These benefits arise from their unique abilities to learn, adapt and innovate as well as to apply expert knowledge to a problem and create new knowledge. These advantages have consequently led to their inclusion in spatial analysis research agendas (NCGIA 1989) and they are seen as a way forward by prominent researchers in the field.

3.4 Development Issues

In Chapter 2 Section 2.5, three software engineering strategies for the construction of SDSS were described:

- Bespoke, integrated system
- Homogeneous modular system (built using custom-built components)
- Heterogeneous modular system (composed of custom-built and off-the-shelf subsystems).

There are many factors involved in deciding which strategy is chosen. However, current trends in computing are increasingly leading developers to adhere to a modular software design route as part of their rapid application development approach. For developing decision support systems, this approach is generally preferred because a wide range of functionality can be easily built into a system without sacrificing key design objectives.

It would be useful to briefly review the development methods used for existing SDSS. Table 3.1 summarises the implementation approach employed and, where applicable, the method used to integrate the system components (either *loose coupling* or *tight coupling* – see Section 3.5 for a full description of these terms).

It is interesting to note that several of the SDSS reviewed use GIS for the mapping and display subsystem coupled loosely with the other system components. This has important consequences for the design of the system as a whole. Clearly, integration of GIS with analytical techniques and models (including Intelligent Systems techniques) and decision support capabilities is therefore a key issue in the construction of ISDSS. These issues are discussed in detail in the next section.

SDSS	Application	Implementation Details	Integration Method
MLDSS (Kohsaka 1993)	Retail store location	Uses custom analysis software (written in MS FORTRAN) and GIS (IDRISI) for 3D mapping	Loose coupling (via files)
EDSS-1 (Guariso and Werthner 1989)	Environmental management	Homogeneous system composed of a database, knowledge base, model base, system manager and dialogue module (GUI). System built using LISP.	None
KBDSS (Armstrong, De et al. 1990)	Locational planning	Homogeneous system composed of a knowledge base, sub-problem-solver and metaplanner.	None
DDSS (Evans and Norback 1985)	Distribution vehicle scheduling	Homogenous system	None
GADS (Carlson, Bennett et al. 1974)	Locational planning	Homogeneous sytem composed of one program for extracting data from a database and another providing analysis and display procedures.	None
UDMS (Robinson 1983)	Settlement planning	Modular homogeneous system composed of 22 separate programs.	Loose coupling (via files)
NYCFP (Blum 1972)	Locational planning	Homogenous system	None
SDSS for Reorganising Service Delivery Systems (Armstrong, Rushton et al. 1991)	Locational planning	Heterogeneous system. Uses linked computers to exchange files between analysis program and GIS (Atlas).	Loose coupling (via files)
SDSS for Location Selection (Willer 1990)	Location selection	Heterogeneous system. Links GIS (TransCAD) with location-allocation modelling package.	Loose coupling (via files)
BANKSIA (Abel, Yap et al. 1992)	Environmental decision support	Modular homogeneous system.	None
ADAPT Davis and Grant 1987)	Land use planning	Homogeneous system.	None
XPLANNER (Han, Kim et al. 1991)	Facility planning and management	Complex heterogeneous system built using a decision support system generator (GURU) . Combines DBMS, expert system modules and standalone optimisation program (Lindo).	Combination of loose coupling (via files) and tight coupling (via procedure calls).
ISIS (Hopkins and Armstrong 1985)	Streams management and planning	Homogeneous system.	None
GIMS (Kessel 1990)	Land resource management	Homogeneous system* based on a suite of programs built around an ASCII-file database.	None

Table 3.1 – SDSS development details

* Efforts are currently underway to loosely couple (via files) GIMS implementations with a commercial GIS (Arc/Info) to provide better mapping facilities (Kessel 1990).

Another observation is that, of those systems which employ a modular approach at the system level, all use loose coupling (i.e. file exchange) to transfer data between the system components.

Of the systems reviewed, three satisfy the ISDSS definition given earlier. Each of EDSS-1, KBDSS and XPLANNER use knowledge based methods to assist in the decision making process by making expert domain knowledge accessible during the problem-solving process.

The following points must be considered during the development of modular ISDSS:

- *Data flows and data interfaces*. Great care needs to be taken to ensure data integrity during any data conversions. This is more problematic for heterogeneous systems.

- *User interface*. Creating a uniform and consistent user interface across all subsystems is a major challenge.

- *Nature of decision support facilities*. As Section 2.3 of Chapter 2 highlighted, one or a combination of three types of decision support may be offered.

- *Incorporation of Intelligent Systems techniques and traditional spatial analysis methods*. A careful assessment needs to be made of the methods that will be used to achieve the design goals of the system. The strengths and weaknesses of each intelligent technique need to be taken into account in order to develop a strategy for applying them to the problem at hand. In many cases it may be necessary to combine the strengths of more than one technique to realise the desired objective. An assessment of the spatial problem-solving abilities of each intelligent technique is provided in Table 3.2.

- *Cost*. Most systems will be built for a very specific purpose and often for a single organisation, resulting in higher development costs.

- *Hardware platform*. This can have important implications regarding the availability of development software, toolkits, class libraries and the integration mechanism employed.

- *Cross-platform support*. If more than one platform is to be supported, a choice needs to be made between maintaining multiple code bases and making use of cross-platform development environments (that commonly abstract the hardware, file system and user interface details of each platform).

Technique	Learning ability	Modelling soft information	Modelling spatial processes	Explanation of outcomes	Handling imprecision
Neural networks	●	●	●	•	●
Genetic algorithms	●	●	●	•	●
Fuzzy logic	•	●	•	●	●
Expert systems	•	●	●	●	•
Rule Induction	●	●	●	●	•

Table 3.2 – Spatial problem-solving properties of intelligent techniques

- *Use of decision support system generators*. Toolkits such as GURU (Micro Database Systems Inc. 1987) are designed for developing decision support systems and offer various tools including in-built programming languages for providing additional functionality.

- *Spatial data visualisation*. A GIS or other display software may be employed for this element. Alternatively display tools can be coded from scratch.

Many of these issues are beyond the scope of this thesis. The issues associated with integration and decision support are dealt with in the next section and subsequent chapters.

3.5 Software Integration Issues

Existing approaches to addressing the problem of integrating modelling methods with SDSS and GIS range from tightly coupled systems to loosely coupled systems. There is inevitably a trade-off in the linking strategy employed. A balance must be struck between offering high speed data access but remaining GIS-specific on the one hand (e.g. using purpose-built external modules) and lower speed access whilst being able to link to a number of different GIS on the other (using file based interaction, for example).

This section explores the possibilities presented by employing client-server technologies. At the simplest level, these are essentially comprised of operating system supported protocols for handling message passing between applications. Examples include

Microsoft's DDE, Apple's Apple Events and sockets for Unix systems. These have the potential of offering high speed data access yet being able to simultaneously link to different GIS systems.

GIS are being used increasingly for spatial analysis and modelling, moving into territories for which they were not necessarily designed. A variety of statistical methods, models and analytical tools have been used for data analysis, decision support and forecasting in applications such as land-use classification and location selection.

This has resulted in a number of *ad hoc* methods of getting around the problem of combining modelling functions with GIS. Often this takes the form of separate external packages that are poorly integrated with the viewing system. Consequently the linking strategies are wholly incompatible with one another, so that an analysis tool built for one GIS will not generally accept data generated by another.

The spectrum of approaches available for integrating analytical tools within GIS has two extremes (Batty and Xie 1994): *Loose Coupling* and *Tight Coupling*.

3.5.1 Loose Coupling

Loosely coupled systems (see Figure 3.1) interact by using files to exchange data between the GIS and the (separate) analysis/modelling package. For example, one might directly read TIGER files into a modelling system and perform some analysis on that data.

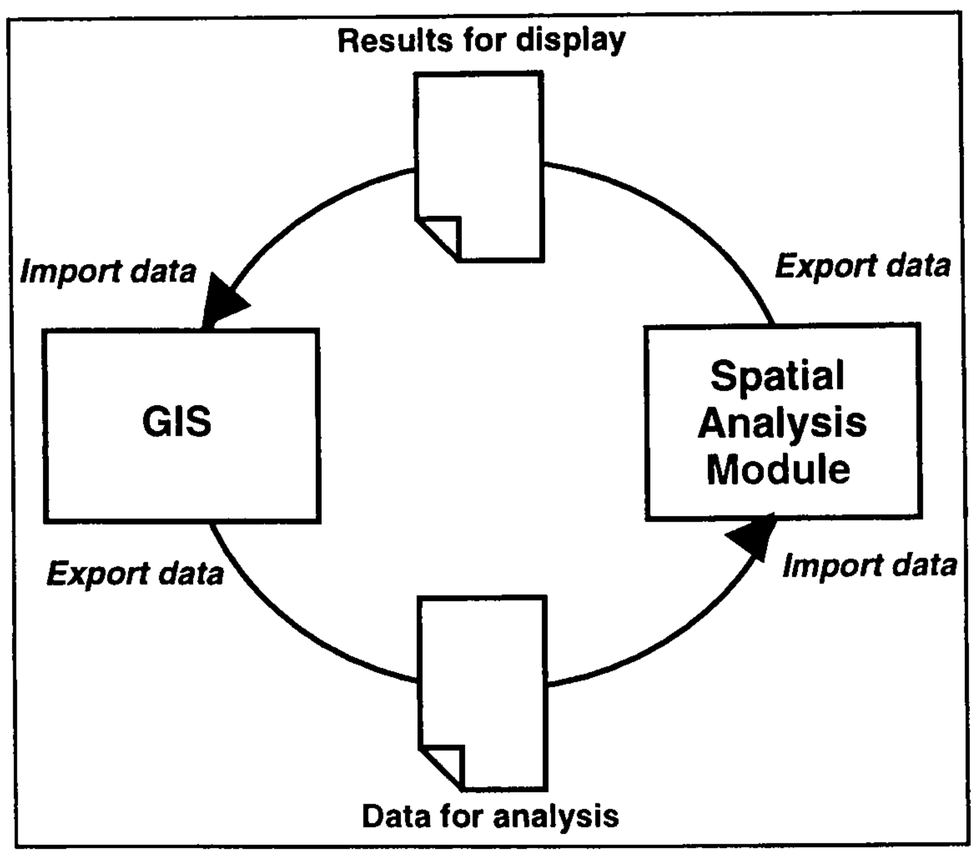


Figure 3.1 – Loosely coupled approach to integration

The advantage of this approach is that the modelling subsystem can integrate with more than one type of GIS by supporting a number of different data formats. Hence, loosely coupled systems can be *non-GIS-specific*. However, this is achieved at the expense of other desirable features. First, the speed of data transfer between the modelling system

and the GIS is much lower, and generally calls for the user to manually export the data of interest before switching to the modelling system to read the data.

Secondly, the level of integration is much lower since interaction is at the file level. This means that it is virtually impossible to implement a common user interface with linked operations. For example, point-and-click operations cannot be supported between the GIS and the modelling program.

3.5.2 Tight Coupling

In *tightly coupled* systems (see Figure 3.2) analytical and modelling utilities exist as specially coded modules within the framework of the GIS. Such modules might be written in a language provided by the GIS (e.g. the Arc Macro Language for Arc/Info) or a standard language like C or C++. This effectively results in an extension of the GIS' functionality and the analysis system can appear to be seamlessly integrated from a user interface viewpoint. For example, point-and-click queries could be supported in such a system, providing much more intuitive answers for users. Here the GIS would dispatch an *(x,y)* co-ordinate pair to the modelling program, which could then display a result within a window.

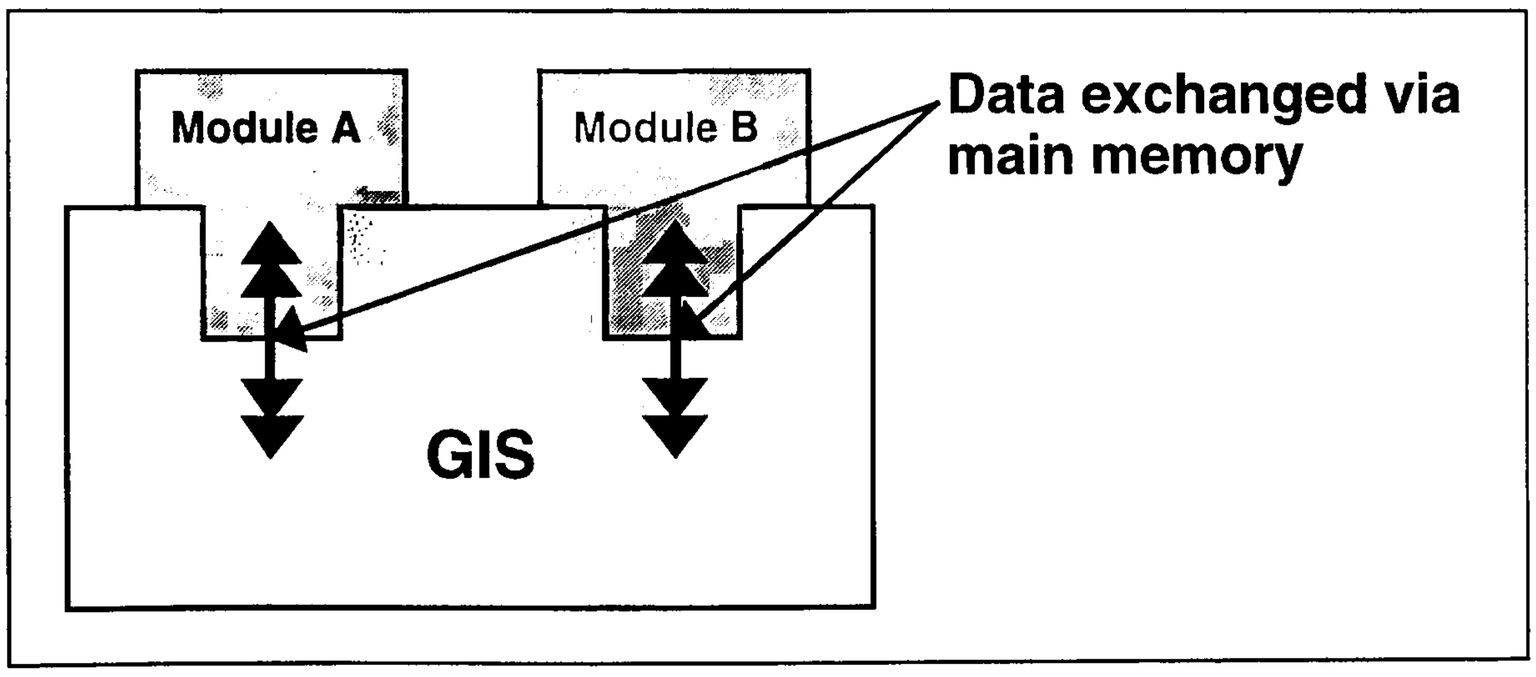


Figure 3.2 – Tightly coupled approach to integration

Speed of operation is a further advantage of this approach, since all data transfer between the GIS and the modelling subsystem can be performed in main memory.

However, modules built in this way are necessarily *GIS-specific*, since other GIS will generally not support a compatible macro programming language or cater for an identical set of user procedure calls (i.e. hooks). Consequently, there is no prospect of being able to run the same models on other GIS. From purely a research perspective, this may not present a problem but in a commercial environment, such compatibility across multiple applications in the same class is highly desirable. For example, the ability to read Lotus 1-2-3 spreadsheets into Microsoft Excel and vice versa is an essential feature both for users,

who inevitably work in a heterogeneous application environment, and application developers, keen to maintain and enlarge market share.

3.5.3 Co-operative Approach

Therefore, using current integration methods a trade-off must be made among the three properties discussed above, namely speed of data transfer, GIS-specificity and level of integration. Clearly an alternative strategy is called for which can allow the development of systems with all three of these properties. Such an approach would enable high speed data transfer and permit tight model and GIS integration whilst being portable across a variety of GIS platforms. This deficiency has been noted in the literature (Goodchild 1991):

". . . still missing is an effective form of tight coupling, in which data could be passed between a GIS and a spatial analysis module without loss of higher structures . . . At present this is impossible . . . "

In reviewing various SDSS's capabilities for performing *visual interactive modelling* (whereby the modelling system interacts closely with visual data representations and other user interface elements for effective exploratory spatial analysis), As Densham notes (Densham 1994), "SDSS supporting VIM [visual interactive modelling] must be tightly integrated from the technical perspective because complex data and command flows are required" and also that "each of the system's modules must be integrated seamlessly with the others." However, only the tightly coupled approach is recognised in the literature as providing this level of integration (Fedra 1993, Nyerges 1993).

A further consideration stems from the observation that end user computer systems are increasingly composed of a set of heterogeneous machines that are connected together via a communications network. For example, a town planning department might have Unix workstations and PCs running on an Ethernet network. Any integration approach adopted should be capable of delivering results despite such a jumble of diverse systems. Obviously, it would be highly desirable if the integration method could in fact utilise these distributed systems for the benefit of end users.

This thesis proposes that client-server technologies can satisfy the above set of requirements. This intermediate route to integrating modelling systems with GIS has been termed the *co-operative* approach (Sandhu and Treleaven 1995), of which two types have been proposed . First, with *direct co-operation*, the modelling system is directly linked with the GIS via IAC as shown in Figure 3.3. Secondly, *indirect co-operation* is characterised by the existence of an intermediate interface between the modelling system and the GIS. A knowledge-based decision workbench would be an example of such an

interface. This classification is useful for relating the IAC approach within the spectrum of coupling mechanisms and for distinguishing the type of IAC interaction implemented.

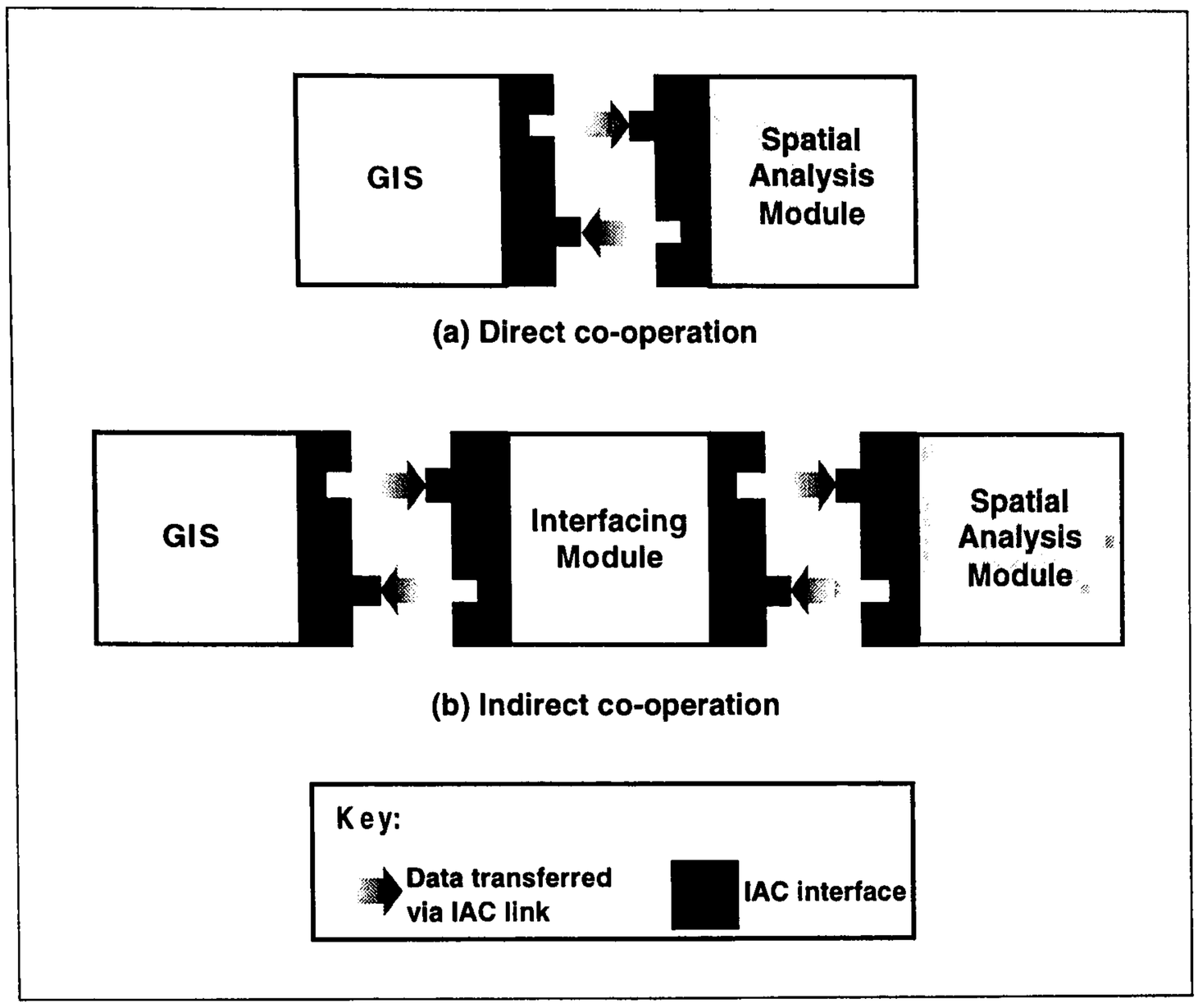


Figure 3.3 – Co-operative Approaches to Integration

The benefits of this approach are described below:

- *Achieves tight, flexible coupling, permitting linked operations.* For example, clicking on a map feature in the GIS could activate the analysis tool to display various computed values for that particular feature. This is an invaluable feature for spatial decision making (Densham 1994).

- *Permits rapid data transfer between system modules.* This is limited only by the system's operating speed and network bandwidth, if the ISDSS is distributed.

- *Allows efficient use of distributed computing resources.* This can be a major advantage for models that are too complex to be run on the same machine as the GIS, such as models for crop growth and atmospheric circulation (Heuvelink, Burrough et al. 1989). Indeed, a SDSS based on two simply linked machines, one for analysis and the other hosting the GIS, has been built to overcome such difficulties (Armstrong, Rushton et al. 1991). However, this system employs loose coupling to integrate the two subsystems using a parallel port link and manual file exchange.

- *GIS-independent modelling tools can be built.* By implementing multiple interfaces on the analytical module, any number of different GIS can be supported.

- *Supports multitasking*. For system modules that are resident on the same machine, this approach naturally provides the messaging infrastructure to support multitasking of the modules. Multitasking allows efficient use of processing resources.

Despite these advantages, the following limitations need to be recognised:

- Few existing GIS support dynamic linking.

- Linked operations require linking both at the data level (i.e. allowing access to data structures) and also at the event level (i.e. the IAC interface must provide hooks into the program's main event loop). This can have serious implications as far as the system's design is concerned.

- Requires additional overhead. Additional processing needs to be done by the system to dispatch messages and data between processes. Therefore, data transfers will be slightly slower compared to those possible via tight coupling.

The validity of the co-operative approach to integration is explored in the construction and deployment of a prototype client-server based ISDSS (GeoAnalyser) in Chapter 4 and 5. Various architectures for ISDSS employing this approach are proposed and their feasibility discussed in Chapter 7.

3.6 Taxonomy

There are many examples of the application of intelligent techniques in spatial problem solving. However, there is currently no framework with which to compare and relate these systems. This section presents a classification scheme for the application of intelligent techniques for spatial decision support. The taxonomy provides a useful framework with which to contrast the architectural and functional roles played by intelligent techniques within the decision support system.

The classification scheme is intended mainly for ISDSS. The numerous examples of intelligent techniques being used for spatial analysis and problem solving, which operate mainly by linking with a GIS, may also be classified using this approach. This is justified because (i) these applications focus on specific development issues that are also relevant to the construction of ISDSS, and (ii) they fulfill some of the criteria used to define

ISDSS. Examples of these systems and fully-fledged ISDSS are used to illustrate each class of the taxonomy.

The application of intelligent techniques for spatial decision-making falls into one of three categories: *Data-oriented Systems*, *Model-oriented Systems* or *Decision-oriented Systems*.

3.6.1 Data-oriented Systems

Data-oriented ISDSS employ intelligent techniques for either data storage, query or data conversion. The techniques may be used to improve the basic characteristics of the spatial data itself, the mechanisms used to store or retrieve it or otherwise alter the data during its flow through the system.

Wang's prototype land assessment system (Wang, Hall et al. 1990) is an example of a data-oriented system. The system links closely with the Arc/Info GIS and uses a fuzzy database model and fuzzy query handler to perform land suitability assessment.

3.6.2 Model-oriented Systems

Model-oriented ISDSS use intelligent techniques to model or analyse spatial data to generate understanding or produce new information as an intermediate stage in the process of developing new knowledge. Examples include using neural networks for forecasting and pattern recognition, fuzzy classification and using expert systems to generate model outcomes.

ESSAS is an example of this type of system. It uses an expert system (developed using Personal Consultant Plus) loosely coupled with a CAD-based display package to select sites that avoid environmental, construction and other problems while at the same time adhering to the various end-user regulations. The knowledge base contains a wealth of rules for determining, suitability, calculating the land area required for building construction and creating maps to provide additional information to the user.

3.6.3 Decision-oriented Systems

In decision-oriented systems, intelligent techniques are used to provide decision makers with a supporting framework for basing their decisions or make available knowledge about procedures, models and techniques. Examples include expert systems to guide the decision making process, assist in the use of complex models, advise on relevant factors involved in data preparation and processing or assessing the quality of solutions. Typically the knowledge base is encapsulated within the main controlling module, closely

tied to the user interface, which forms a shell for problem exploration, providing facilities for the execution of models and evaluation of outcomes.

KBDSS is a decision-oriented ISDSS that uses frame-based rules to encode:

- descriptive knowledge about the problem-domain,
- heuristics on solving locational problems, and
- knowledge about data relationships (speeds up analysis).

The system supports decision making in the following ways:

1. Determining the nature of the problem.
2. Determining the nature of a suitable solution.
3. Finding the solution, using a library of problem solving methods.

The applications discussed above and additional examples of the taxonomy are summarised in Table 3.3. There are also many research examples applying intelligent techniques to address specific spatial analysis issues, such as using genetic algorithms to automatically generate spatial interaction models (Openshaw 1988) and using expert systems to classify land-types from remotely sensed data (Wharton 1987).

3.7 Summary

This chapter has created a foundation for the development of Intelligent Spatial Decision Support Systems as a distinct class of system for spatial decision support. Using a definition for ISDSS as a starting point, deficiencies in existing SDSS have been highlighted. It has been established that there are three key ways in which intelligent techniques can address some of these deficiencies: enhancement of spatial modelling and spatial data analysis, knowledge processing and enhanced decision support.

Next, issues surrounding the development of ISDSS have been presented with reference to existing systems. It was found that modular systems (employing GIS for display purposes as well as other external tools) mainly use loose coupling as the means to exchange data between the modelling/analysis subsystem and the display subsystem. Software integration issues were discussed separately and, on assessing existing methods of integration, it was found that there is a need for a fast, flexible means for the exchange of data structures and control between subsystems in order to create decision support systems that allow highly interactive exploration of the problem domain. It has been proposed that client-server methods may be capable of serving such a purpose. The

benefits and limitations of what has been termed *co-operative* coupling have been discussed.

System	Description	Method	Class
ADAPT (Davis and Grant 1987)	Creates zoning schemes by using knowledge based methods to determine activity allocation.	Rule-based systems	Model-oriented
XPLANNER (Han, Kim et al. 1991)	Uses expert system to store domain specific knowledge for facility planning and management. Uses several knowledge bases for (i) estimating parameter values for an optimisation model, (ii) screen existing facilities, and (iii) interpreting the results of the optimisation model. Also uses rule induction for creating decision rules.	Expert systems	Model-oriented, Decision-oriented
KBDSS (Armstrong, De et al. 1990)	Uses rules to store (i) descriptive knowledge about the problem-domain, (ii) heuristics on solving locational problems, and (iii) knowledge about data relationships (speeds up analysis).	Rule-based systems	Decision-oriented
Land assessment system (Wang, Hall et al. 1990)	Prototype system links the Arc/Info GIS with a fuzzy database model and fuzzy query handler for performing land suitability assessment.	Fuzzy logic	Data-oriented
Tourism information system (Wang 1994)	Prototype system uses a fuzzy preprocessor to query an Arc/Info database.	Fuzzy logic	Data-oriented
ESSAS (Han and Kim 1989)	Uses an expert system (loosely coupled with the CAD package AutoCad) for site selection and analysis. Applies rules to assess the suitability of individual sites.	Expert systems	Model-oriented
MAP-AID (Robinson and Jackson 1985)	A map design system that uses an expert system (linked with the GIMMS mapping package) to implement map design rules and provide an intelligent user interface.	Expert systems	Decision-oriented
RESHELL (Goodenough, Goldberg et al. 1987)	An expert system shell for matching remotely sensed data with map features. Uses knowledge bases for (i) map-image congruency rules and (ii) controlling system runs (obtaining the problem specification and data, running congruency expert, assessing results and generating output.	Expert systems	Model-oriented, Decision-oriented
Map management system (Goldberg, Goodenough et al. 1985)	Uses expert systems to automatically update forestry maps (stored in an Intergraph GIS) based on Landsat data. The hierarchical expert system stores knowledge on the strengths of the various image processing algorithms, details of function arguments and rules for interpreting the results.	Expert systems	Model-oriented
LEES (Wei, Jianbang et al. 1992)	Uses an expert system loosely coupled with a GIS (Arc/Info) for performing land evaluation.	Expert systems	Model-oriented
Land assessment system (Wang 1994)	Uses a neural network (loosely coupled to Arc/Info) for land suitability assessment.	Neural networks	Model-oriented

Table 3.3 – Classification of intelligent techniques for spatial problem-solving

A taxonomy for the classification of ISDSS has been developed which helps to assess the functional and architectural roles played by intelligent systems. The taxonomy was used to characterise various applications of intelligent techniques for spatial problem-solving.

Chapter 4

The GeoAnalyser Intelligent Spatial Decision Support System

This chapter describes the design and implementation of the GeoAnalyser ISDSS. The design rationale, user interface and system components (covering the display tools, model controls, spatiotemporal database, genetic algorithm and spatial interaction model) are presented in detail. The method used to integrate the distributed system components is discussed in depth. The decision support features of GeoAnalyser are also introduced.

4.1 Introduction

In Chapter 3, a foundation was laid for Intelligent Spatial Decision Support Systems. Various issues surrounding their development were discussed and a classification scheme was presented. Several of these issues are explored further in this chapter and also in Chapter 5, which uses GeoAnalyser to explore the dynamics of high technology industry in the South-East of England.

The following aspects of ISDSS development are explored with the aid of the GeoAnalyser system:

- Integration of system modules through the use of *direct co-operative coupling*.
- Provision of decision support facilities through an interactive user interface and spatial model.
- Spatiotemporal data visualisation.
- Decision support tools capable of handling spatiotemporal data.

4.2 Design Objectives

GeoAnalyser is a prototype ISDSS designed specifically for the interactive exploration of spatiotemporal data. It was developed to fulfill a need within a management consultancy company (KPMG Peat Marwick, High Technology Practice) for a system which could be used to visualise spatiotemporal data (i.e. data that has both a spatial extent and a temporal extent) and deploy a *spatial interaction model* describing the behaviour of the

spatial system of interest to the decision maker. Such models can be used to shed new light on spatial phenomena and perform what-if analysis and forecasting.

The following are the key requirements of the system. Existing systems fall considerably short of these requirements.

- *Provision of spatiotemporal visualisation tools*. To help understand the evolution of a system through time, good visualisation tools are vital. However, current systems are only designed to generate static two dimensional maps.

- *Provision of generic tools for developing, calibrating and testing spatial-interaction models*. The creation of a spatial interaction model is a complex task. The system should provide support for both their development and deployment. Such models can theoretically be developed within a GIS, but due to their lack of an infrastructure for handling temporal data, this route is not very effective.

- *Support for temporal data*. Spatial time series data is the main starting point for developing spatial interaction models. However, GIS and SDSS only provide facilities for handling the spatial component of data – the temporal component cannot be accommodated in the data models of current systems. Fortunately, this issue is being addressed in new research (see Yuan 1996 for a summary).

- *Fast access to spatial data structures*. Complex spatial interaction models (such as those modelling migration effects) require many thousands or even millions of operations for each model run. This requires rapid access to all relevant spatial data. However, GIS rely on (relatively) slow database operations to store and retrieve spatial data. This is exacerbated by the use of the relational data model (Codd 1982) in some implementations for representing intricate data structures. Such structures, split across many tables, require several 'join' operations for their retrieval. Thus complex models experience serious speed problems if they rely on GIS for the provision of data, such as when executed as modules within the GIS.

- *Exploitation of distributed computing resources*. Although not a direct requirement, it would be advantageous if the system can make use of networked processing resources, especially since spatial models are so CPU hungry.

GeoAnalyser was developed in response to these requirements. The shortcomings of existing systems was the deciding factor in choosing to build the system as a homogeneous modular system. Nevertheless, GeoAnalyser can work alongside a GIS to make use of the latter's superior mapping facilities.

The key end user requirement of GeoAnalyser is to provide a black box tool for assessing the effect of policy measures on the spatial system being studied (via the use of a spatial interaction model). In the case of high technology industry, such measures include tax incentives and interest rates. A broad conceptual overview of this use of GeoAnalyser in a given environment is shown in Figure 4.1.

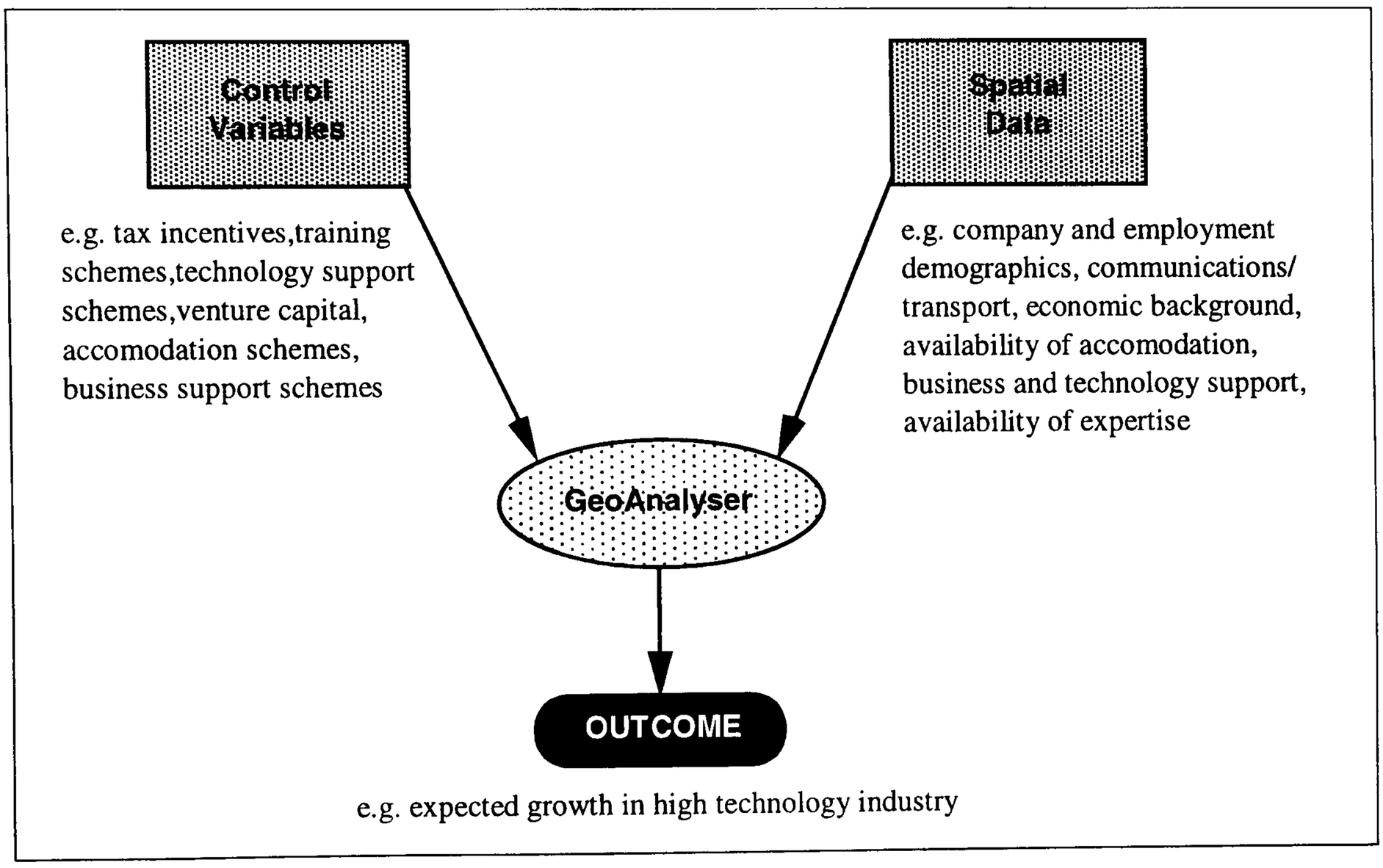


Figure 4.1 – Conceptual overview of GeoAnalyser

4.3 System Overview

The implementation of GeoAnalyser follows a two-tiered approach as shown in Figure 4.2. *Direct co-operative coupling* is used to link together the two parts of the system: *GeoServer* (containing the spatial interaction model and fitness evaluation function for the genetic algorithm) and *GeoClient* (containing model and data management, display tools, genetic algorithm and simple data analysis functions). This separation of the user interface from the modelling module has the following benefits:

- *Design and implementation autonomy*. The model can be developed independently of the main program provided the public data and command interface specified for the server is complied with.

- *Utilise distributed processing resources*. Most workplace computer systems are used in networks where, with the inevitable adoption of new faster machines in the due course of time, the system becomes a heterogeneous mix of older machines (e.g. 486 PC or 68000-based Macintosh) and newer machines (e.g. Pentium Pro, Power Macintosh or multiprocessor machines) exhibiting a wide range of processing power. Furthermore, for specialised CPU-intensive applications, such as modelling or database servers, most systems managers tend to rely on the latest, top-of-the-range machines. By allowing the most processing intensive parts of the decision support system to be run from any machine on the network, the ISDSS can make better use of these distributed resources.

- *Parallel model execution*. Offers the potential to run several models independently using different machines to host each GeoServer application. This feature has not however been developed in the prototype system.

The model object, as shown in Figure 4.2, is a *plug-in module* which is loaded when GeoServer is run. Apart from the obvious benefit of allowing different types of model to be run within the same execution environment, this plug-in architecture enables model developers to test slightly different versions of the same model to compare accuracy and goodness of fit.

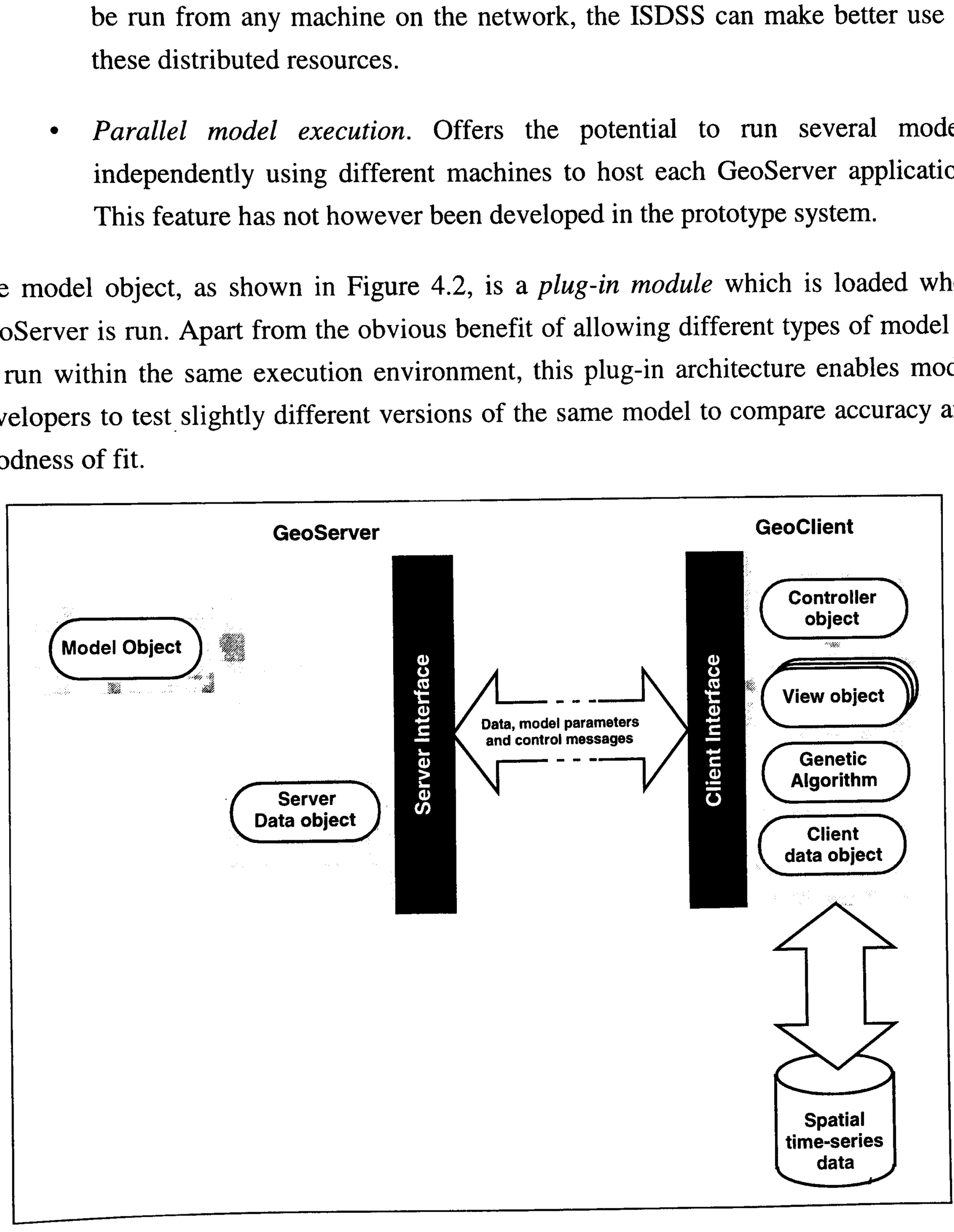


Figure 4.2 – Design overview of GeoAnalyser

GeoAnalyser has been implemented to run on Apple Macintosh computers and makes extensive use of the Mac OS Graphical User Interface (GUI). As a result, the system is relatively easy to use with the minimum of training. The system was implemented in C++, making use of it's high execution speed and support for object-oriented programming. The Think Class Library, a commercial C++ class library for the construction of GUI elements, was used to implement standard and custom user interface objects.

A sample GeoAnalyser screen display is shown in Figure 4.3.

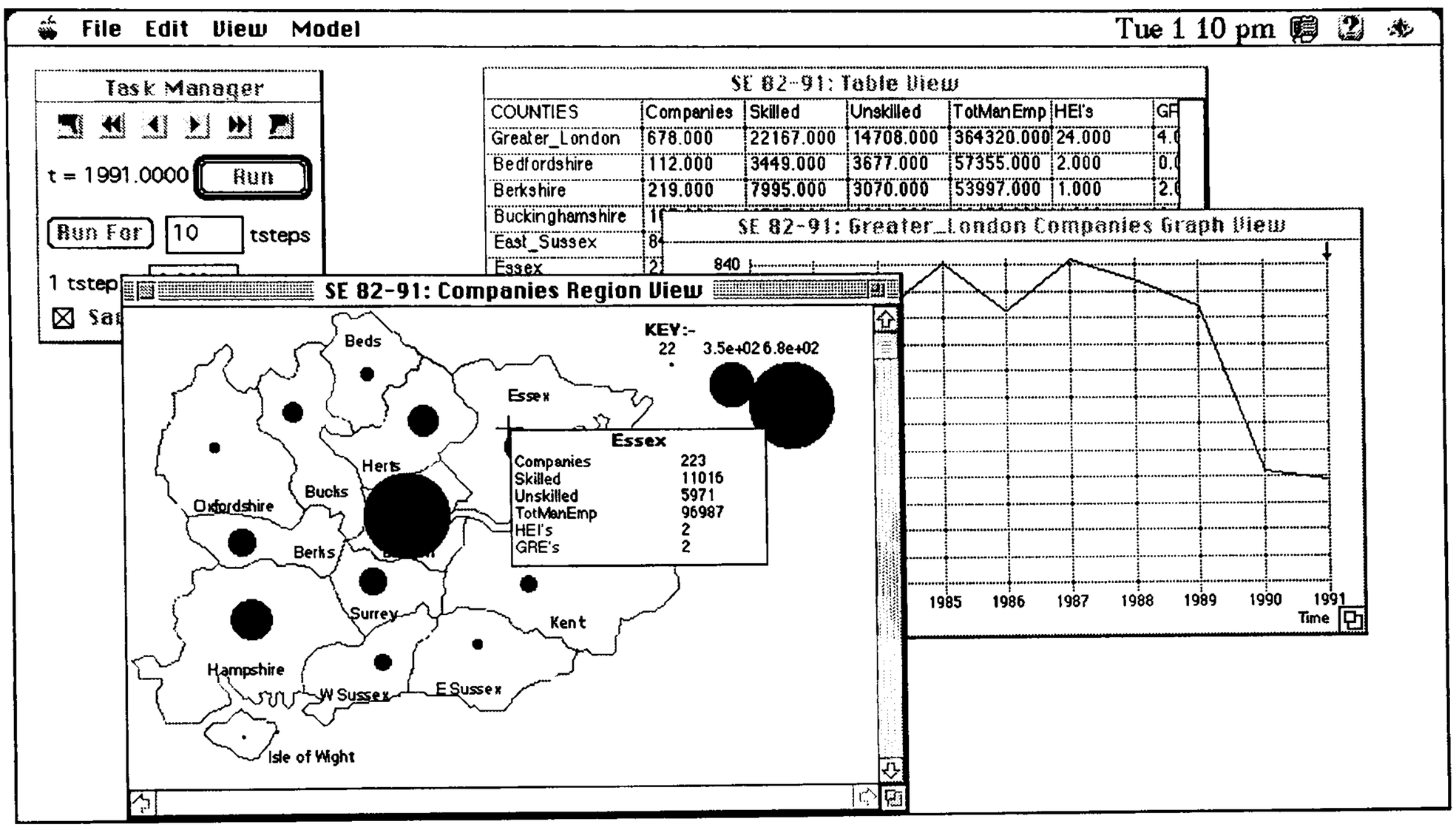


Figure 4.3 – Sample screen display of the GeoAnalyser system

Key design and implementation features of GeoAnalyser are discussed in the subsections below, covering *System Design*, *Data Management*, *Visualisation Tools* and *Integration Methodology*.

4.3.1 System Design

Object-oriented design principles were used to construct the system as shown in Figure 4.4. *Encapsulation* ensures that each of the objects are self-contained and can be developed using black box methods. *Inheritance* allows common behaviour to be passed on from one object to another, thus preventing unnecessary code duplication. *Polymorphism* permits a single object to inherit behaviour from multiple parent objects.

Each of the system objects accesses data via a *Data Manager*. The local data manager (i.e. on the client side) refers all data requests or changes to the remote data manager (on the server side). Upon initialisation all data is read from disk and loaded onto the server

via the local data manager. This strategy isolates all code relating to client-server interaction to just two objects. In fact, this set-up was actually evolved from an earlier non-client-server version of the system where all data resided in the local data manager – the object-oriented approach ensured that no code changes were required elsewhere in the system.

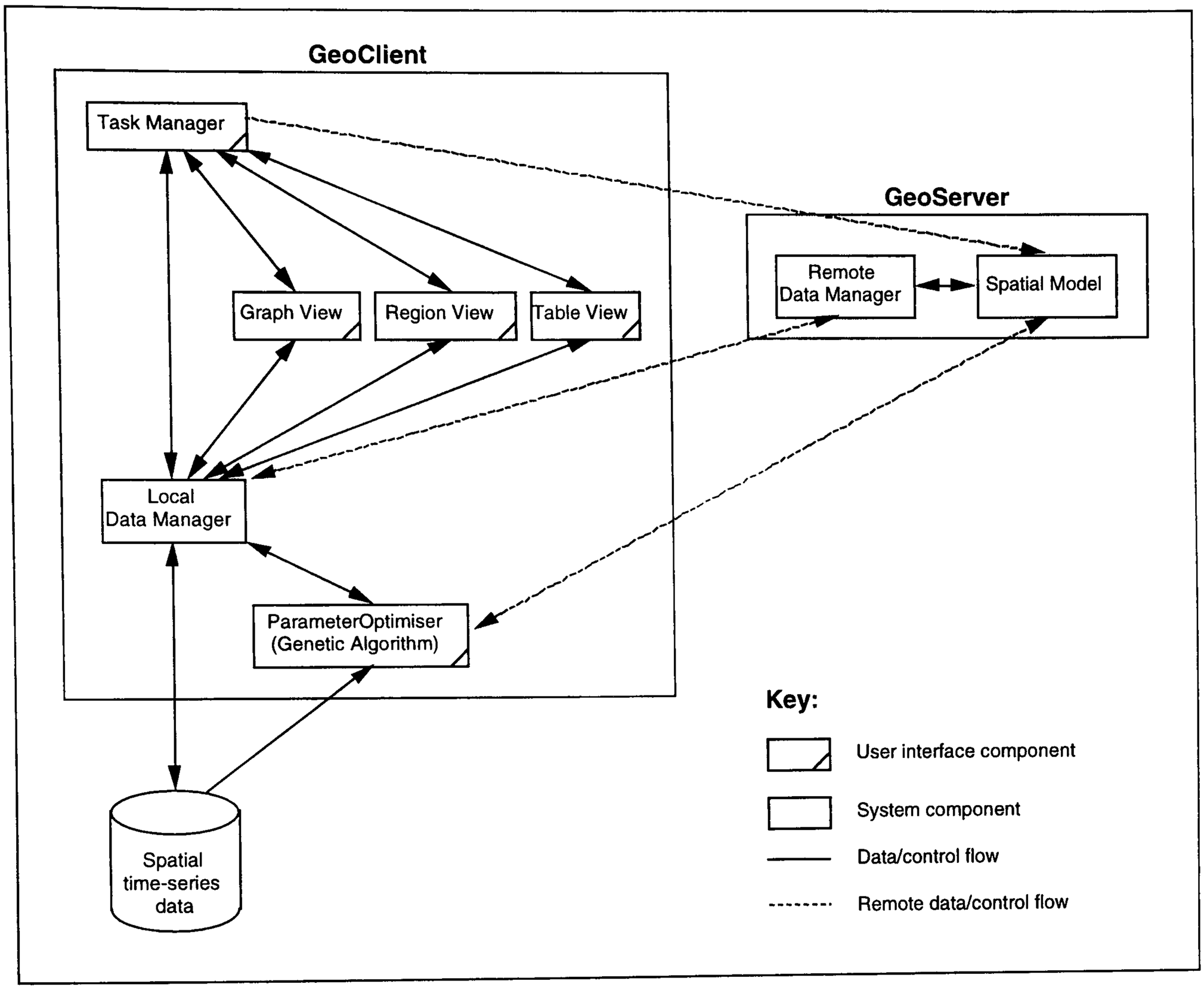


Figure 4.4 – Operational overview of GeoAnalyser

In addition to these data flows, control messages ensure synchronisation of views. Thus, each of the view objects are linked so that changes to the underlying data are reflected in the representation on-screen.

Various system and model settings are accessed using either pull down menus (not shown in the figure) or the *Task Manager* which provides controls for stepping through time and interacting with the spatial model.

4.3.2 Data Management

Extensive support is provided within GeoAnalyser for handling temporal data, which is considered to be an intrinsic property of the regional data. The temporal component is represented as a linear sequence of discrete spatial frames (i.e. a linked list). Each frame

has a specific time associated with it. This type of representation is particularly suited to discrete event simulation models (e.g. spatial interaction models).

For the sake of portability, all data is stored in Microsoft Excel-compatible text files. These are read into main memory during initialisation. This simple but highly effective data management strategy has the benefit of offering almost instantaneous access to time series data and portability through the use of an industry standard file format. The former was considered essential for running the model with a fair degree of accuracy in real time. An obvious but serious argument against this use of main memory is that it puts rather a low limit on the amount of data that can be handled at any one time. However, this approach is justified for three key reasons: (i) the speed that is achieved as a result; (ii) the ability to use the modelling features on-line rather than off-line, as is the case with most GIS; and (iii) the upper RAM limit on the current generation of PC hardware is very high (RISC-based Power Macintoshes costing under £1000 are expandable to 264 MB, for example). Since spatial modelling is a specialist application, using machines with a large memory capacity is the only real way of delivering the performance that is required by computing-intensive spatial modelling techniques. A parallel might be drawn here with commercial computer-based graphic design, a memory and processing intensive application area where desktop machines with at least 128MB of RAM the norm due to the size of modern high resolution colour images (which are typically measured in the tens of megabytes). Nevertheless, the approach adopted here will not cater for *very* large datasets without some form of disk-based database.

After initialisation, the data manager in the server contains a *linked list of frames* consisting of time stamped spatial data (see Figure 4.5). Each frame is composed of a time value, a two-dimensional dynamic array containing spatial data (i.e. a set of one-dimensional arrays each representing the spatial variables associated with each geographic point) and a pointer (i.e. link) to the next frame in the time sequence (or *NULL* if this is the last frame). The linked list provides reasonably fast access to the data and naturally preserves the sequence of frames without additional complexity. The use of dynamic arrays permits the dimensions of the spatial data (i.e. number of spatial variables and number of geographic points) to be determined 'on-the-fly' at run time rather than compile time (as is the case with standard arrays).

The following are the public methods (i.e. member functions) of the Data Manager classes (the interface is identical for both the local and remote Data Managers):

Boolean NextFrame(void)

Go to the next frame in the sequence.

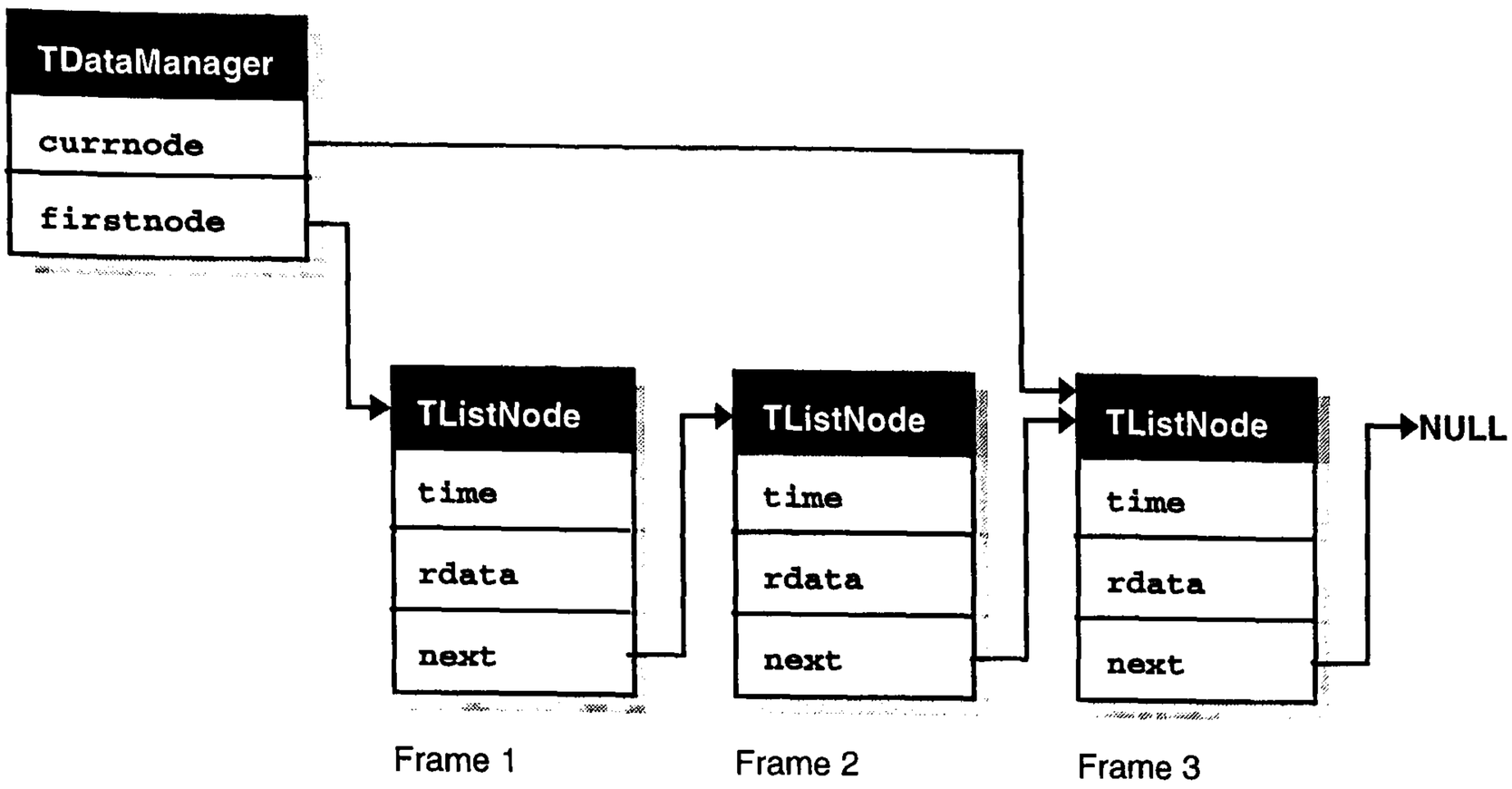


Figure 4.5 – GeoAnalyser Data Storage Model

Boolean PrevFrame(void)

Go to the previous frame in the sequence.

double GetTime(void)

Get the time associated with the current frame.

TRegion GetData(void)

Get all the spatial data stored in the current frame.

double GetDataItem(unsigned short v, unsigned short l)

Returns the value of the specified variable for the specified geographic point.

void AddFrame(double t, TRegion r)

Appends a new frame to the current frame, initialises the new frame using the given time and spatial data and makes it the current frame.

void SetFrame(double t, TRegion r)

Sets the data for the current frame.

Most of these functions provide straightforward access to the data. The exceptions are *AddFrame* and *SetData* which operate so as to *preserve the temporal integrity* of the data set. For example, if the current frame is somewhere in the middle of the sequence, a call to *AddFrame* will create a new frame after the current frame and truncate all frames after this. Similarly a call to *SetData* will also truncate frames after the current frame. This truncation ensures that some measure of continuity and time ordering is maintained in the temporal sequence. Clearly a degree of care needs to be exercised in calling these two functions as they can cause large irreversible changes to the dataset.

4.3.3 Visualisation Tools

The main user interface components consist of the *Task Manager* (described below) and the following three basic display forms (see Figure 4.3):

- *Thematic maps*. These allow viewing of the spatial distribution of variables. Each variable is displayed as filled circle of variable size for positive values and an outline circle for negative values. A number of features are provided to enhance the use of the view. An automatic key is also shown in the top right of the window. As the key is automatically adjusted, a menu option provides a means for locking the scale used to draw the view, allowing visual comparisons to be made over a number of timesteps (by going backwards and forwards in time). The display may be zoomed in and out to highlight areas of detail. Clicking and holding down the mouse button near a plotted point (the cursor automatically changes to a cross shape when it is close to a point) shows a handy pop-up box (as shown in Figure 4.3) containing values for all the variables at that location at the current time.

- *Tables*. These display the raw data in a spreadsheet format. Double-clicking on a cell allows it's contents to be modified via a dialog box.

- *Graphs*. These show a time trace for a particular variable over the entire data set, allowing the time evolution of individual variables to be observed. The active frame is indicated by an arrow above the graph. The scale of the graph and its grid lines are dynamically determined based on the range of the specified data and the size of the window. The cursor keys on the keyboard are activated when the graph view is the active window. In conjunction with the SHIFT key, the left-right keys change the variable that is plotted for the current spatial point. Similarly, the up-down keys change the current spatial point (keeping the variable the same). This provides a convenient shortcut to other variables and other locations within the dataset.

Any number of these views may be opened at once, with each one showing different aspects of the dataset. Each of the views may also be resized and moved as desired. The contents are either automatically resized (for graphs) or scroll bars are activated (for thematic maps and tables) as required.

The thematic and graph views also support the display of simple algebraic expressions involving the spatial variables (calculated over each spatial point). First time differences and percentage changes of the variables can be displayed in the thematic and graph views. However overlay and buffering operations are not supported.

The *Task Manager* (detailed in Figure 4.6) provides VCR-style controls for:

- stepping forwards and backwards in time in single steps;
- jumping to the first or last frame in the current sequence;
- playing a simple movie of the spatial data by automatically stepping rapidly through to the first or last frame.

The animation function has been noted for its value in visualising spatiotemporal data. Dorling and Openshaw have used off-line movies (created frame-by-frame and then saved separately to disk; these were later played back with different software) to investigate the incidence of childhood leukaemia in northern England (Dorling and Openshaw 1992). GeoAnalyser represents the first attempt to embed an animation system within a SDSS.

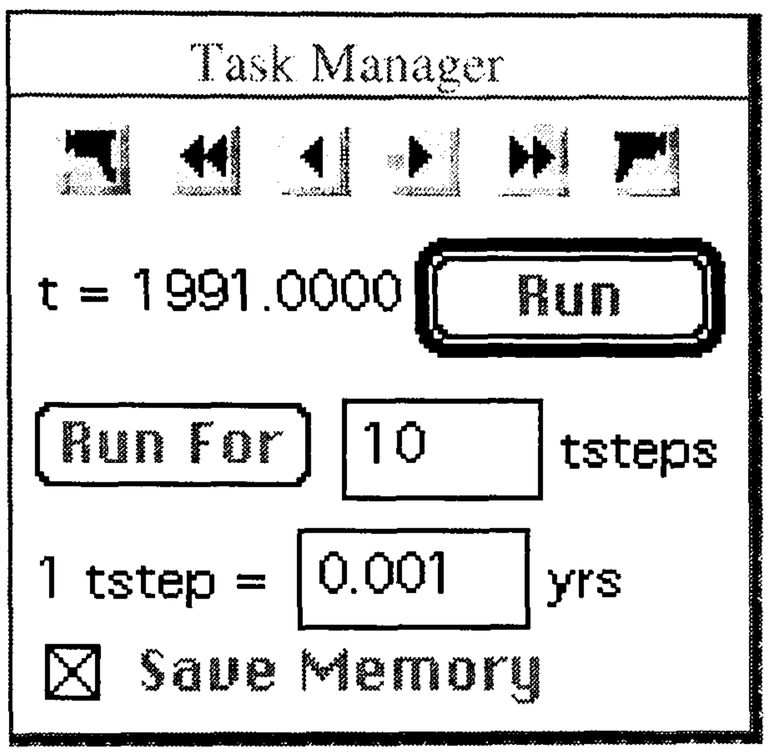


Figure 4.6 – Task Manager Window

The Task Manager also provides controls for executing the spatial model and specifying the model timestep to be used. *Run* executes the model just once, creating a new frame for the generated data. *Run For* executes the model for the specified number of iterations, creating multiple new frames or, if the *Save Memory* checkbox is not set, just a single frame containing the final result of the multiple iterations.

Each of the operations of the Task Manager result in any opened views being updated automatically to display the new data. For general use this does not cause any slow down of the system. However, animating the thematic view was problematic as it took considerably longer to draw than the other views, destroying the effect of a movie. To get around this problem, a *caching scheme* was employed for the thematic view. This initially stores the static elements of the view as a bitmap in memory. Redrawing then involves rapidly stamping the bitmap into the window first and then drawing the dynamic elements of the image on top. Also in the thematic view, the scales locking feature allows the

current scaling of the plotted values to be fixed so as to enhance the perception of change across the view as the movie is played forward or backward.

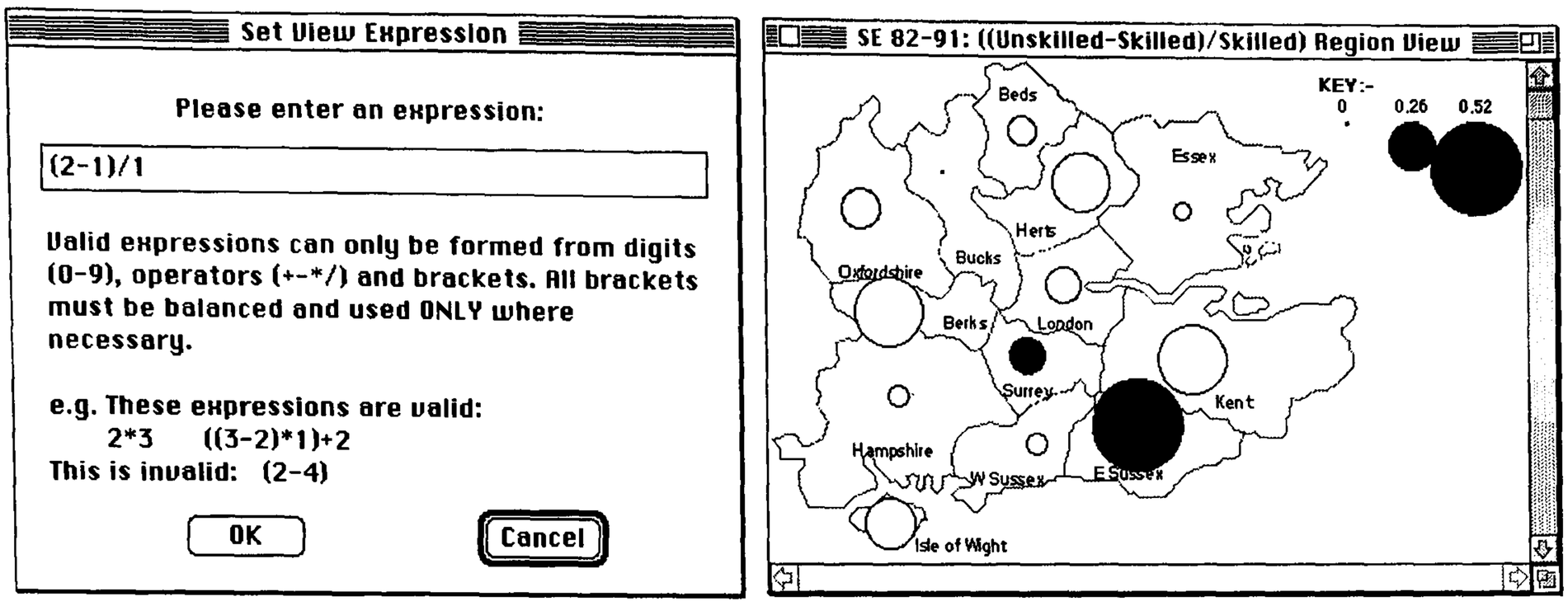


Figure 4.7 – Setting View Expressions in GeoAnalyser

Several time-series analysis tools have also been built into GeoAnalyser. Each of these modify the behaviour of the visualisation tools. First, views incorporating *arithmetic combinations* of the spatial variables may be constructed. An arithmetic combination can be specified for either the thematic or graph view and is constructed using the dialog box shown in Figure 4.7 (accessed from the menu bar). For example, to appreciate what happens to the level of, say, skilled labour compared with unskilled labour, a new view can be created based on the ratio of these two variables. This allows simple analysis of variables to be performed without recourse to a spreadsheet or a special statistical package. This feature is particularly useful when applied to a graph view to observe the time evolution of an arithmetic combination.

Secondly, first differentials of spatial variables can be plotted in either the thematic or graph views. This allows the planner to observe underlying trends in the spatial data and is useful for spotting linear, exponential and polynomial types of growth and decay in the spatial data. Naturally such a feature is vital for being able to understand the time complexity of the spatial system.

Finally, an option in the menu bar displays a window for accessing and modifying the parameters for the spatial interaction model (see Figure 4.8). Pressing the *Optimise* button in this window provides access to the Genetic Algorithm.

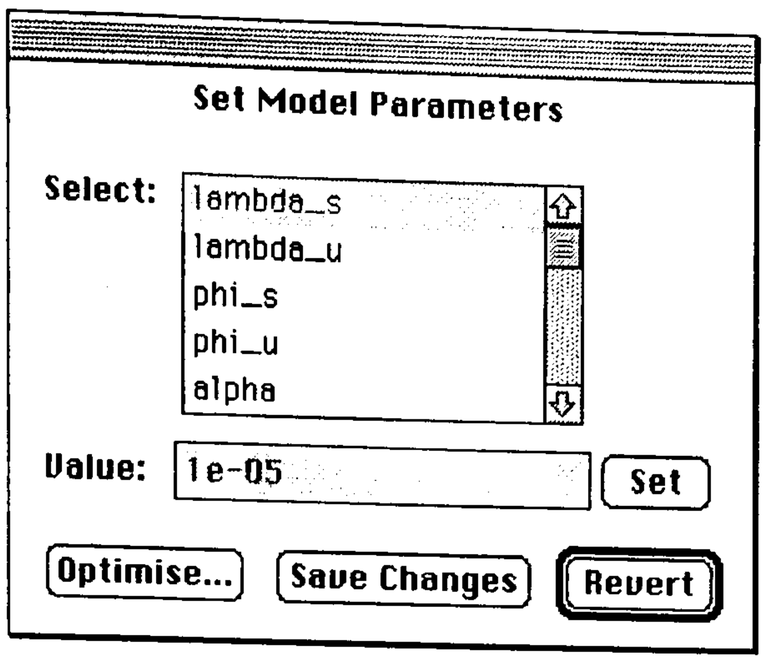


Figure 4.8 – Window for Setting Model Parameters

4.3.4 Integration Methodology

As described in the overview, the design of GeoAnalyser involves a division into two distributed applications: GeoClient (which provides a graphical front end to view data, control model execution and run the Genetic Algorithm) and GeoServer (which provides a plug-in shell for spatial models, the remote Data Manager and the fitness evaluation function used by the Genetic Algorithm). The two applications are integrated using *direct co-operative coupling*.

Currently, GeoClient is designed to run on older Macintosh computers (i.e. those with MC68000 processors). The more processing-intensive GeoServer runs on the newer Power Macintosh range (with RISC-based PowerPC processors). They can both be run separately over a network or, if required, on the same machine (thereby negating the benefit of the distributed architecture).

After selecting a project to work on in GeoClient, a connection must be made with a GeoServer on the network. The dialog box shown in Figure 4.9 is displayed showing all accessible computers on the network on the left hand side and GeoServer applications on the right. Upon selecting a GeoServer to connect with a standard network log-in screen requires the user to enter a valid username and password for the machine hosting the server. Upon successful login, the Task Manager appears and the application is ready to use.

All communication between GeoClient and GeoServer is handled in the form of Apple Events (described in Section 2.5). These allow code to be executed on the server end with data being sent as a parameter block. Similarly results are returned in the same format. This requires data structures (e.g. a frame) to be packed/unpacked into/from a single contiguous memory block. Table 4.1 shows the Apple Events that GeoServer responds to. Most of these directly correspond to the Data Manager member functions described in Section 4.3.2. Model fitness calculations for running the Genetic Algorithm are handled

on the server end (reducing the computational load on the client) by the Apple Events *InitialiseFitness* and *GetFitness*. Four additional Apple Events for frame operations (*SetMark*, *GotoMark*, *GotoStart* and *GotoEnd*) are also included for convenience and to help minimise unnecessary client-server messaging for operations that involve scanning through a large number of data frames. For example, the following pseudocode procedure is executed when opening a new Graph view:

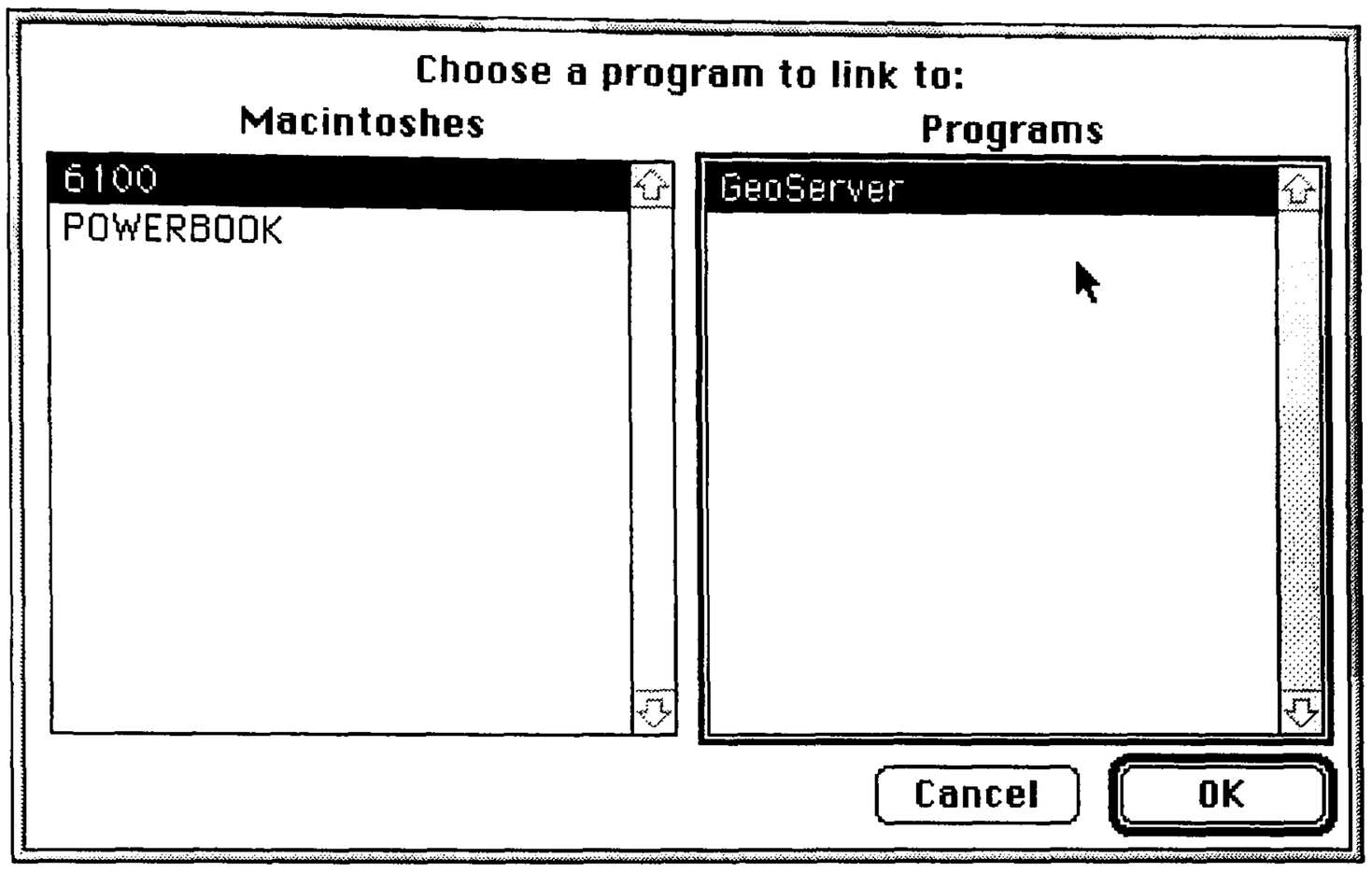


Figure 4.9 – GeoServer Browser Dialog Box

```
SetMark                          //mark current frame
GotoStart                        //go to first frame in dataset
GetDataItem                      //get value to be plotted
WHILE(next data frame){          //do for each successive frame
     GetDataItem                 //get value to be plotted
}
GotoMark                         //go to last marked frame
```

Using *SetMark* and *GotoMark* saves having to use *NextFrame* many times just to return to the active frame.

4.4 Genetic Algorithm

One of the greatest difficulties in building spatial models is that they are very difficult to calibrate. Usually this task may take longer than devising the model in the first place, requiring many runs of the model and the evaluation of the results.

A novel approach is taken with GeoAnalyser to address this problem. Calibration is assisted within GeoAnalyser through the use of a genetic algorithm (GA). This technique, described in Section 2.4, is based loosely on the processes of evolution and natural

selection. The GA provides a means to simultaneously search and optimise the model parameters to *evolve* the most representative set of parameters. The particular GA that is used is an adapted form of QGAME (Dekker 1995)

Operation	Arguments	Description
MODEL OPERATIONS		
RunModel	Time step	Run the spatial model once
RunModelMultiple	Time step, no. iterations, save option	Run the model iteratively
GetModelParameters	–	Retrieve model parameters
SetModelParameters	List of model parameters	Set the model parameters
InitialiseFitness	–	Initialise the fitness functions
GetFitness	List of model parameters, time step	Calculate fitness of model parameters
FRAME OPERATIONS		
GetFrameData	–	Retrieve all data of current frame
SetFrameData	Frame time, spatial data	Set data for current frame
GetFrameTime	–	Retrieves time of current frame
GetDataItem	variable specifier, location specifier	Retrieves variable of a location
NextFrame	–	Set current frame to next frame
PreviousFrame	–	Set current frame to previous frame
AppendData	Frame time, spatial data	Append frame using given data
SetMark	–	Mark current frame
GotoMark	–	Set current frame to last marked frame
GotoStart	–	Set current frame to first frame
GotoEnd	–	Set current frame last frame
MISCELLANEOUS OPERATIONS		
GetPositions	List of positions	Get location positions
SetPositions	–	Set location positions

Table 4.1 – GeoServer Apple Events

The GA can be invoked at any stage during the use of the system via the parameters window. This opens up the main GA configuration dialog shown in Figure 4.10. This configures the number of generations that are to be run (each generation represents a single life cycle in the GA), the number of pools (the GA allows more than one pool to be evolved simultaneously, providing operators to copy or move individuals from one pool to another), the number of chromosomes in each pool and the model timestep to be used. This last parameter determines the timestep that will be used to run the spatial model

when evaluating a set of model parameters. The setting of this value is critical and involves a trade off between a high degree of model accuracy (achieved by using a short timestep, thereby running the model many more times) and the operating speed of the GA. In practice it is best to start with a fairly large timestep for a trial run to ensure that the GA is converging on a solution and then set a shorter timestep in order to run the GA over an extended period of time (overnight, say).

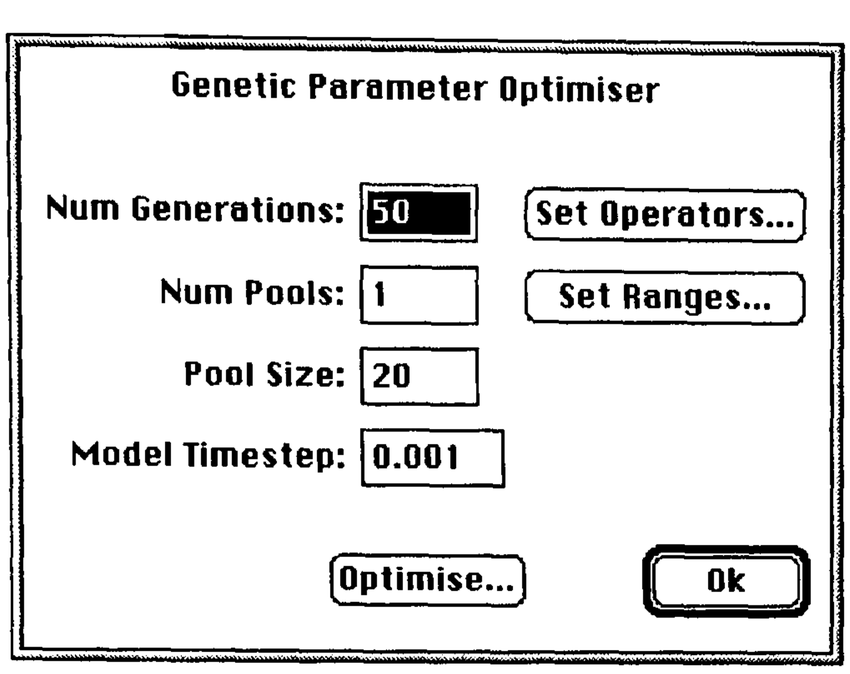


Figure 4.10 – Main GA Configuration Dialog

The main configuration dialog also provides buttons for accessing the *operators dialog* (shown in Figure 4.11) and the *ranges dialog* (shown in Figure 4.12). The operators dialog specifies the genetic operators that are to be used by the GA, covering *initialisation* (initialises the population of chromosomes), *selection* (used to determine the chromosome selection scheme used), *mutation* (mutates genes on a single chromosome), *crossover* (combines two parent chromosomes to create altered offspring), and *migration* (determines how chromosomes are moved or copied from one pool to another) operators and associated *rates* (which determine how much of an effect an operator has).

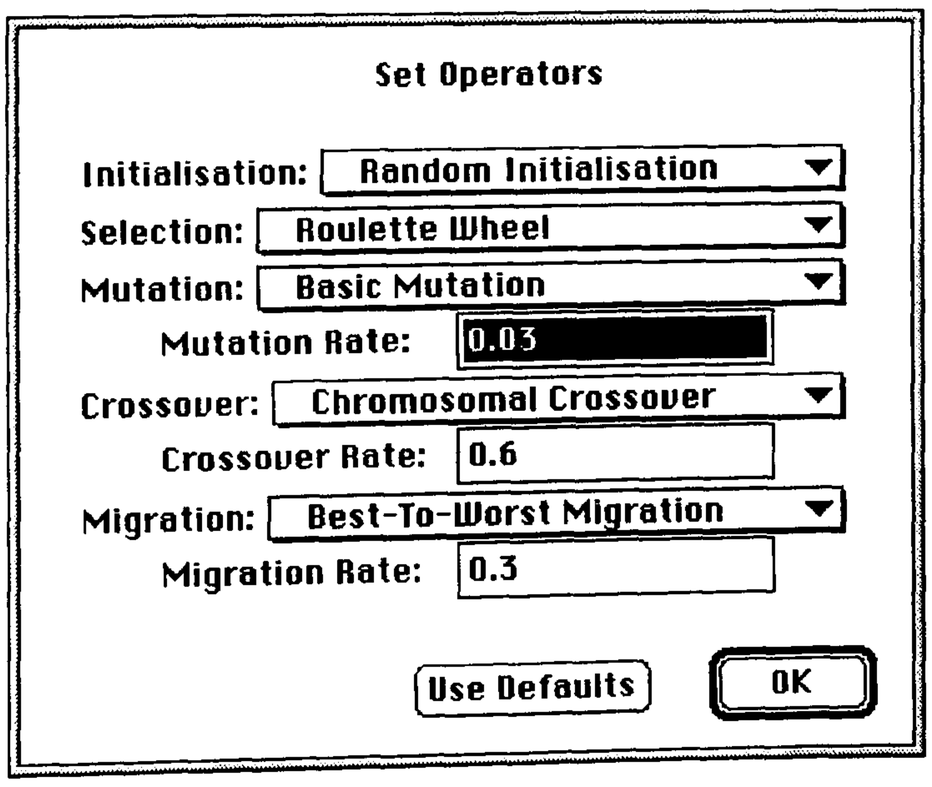


Figure 4.11 – Genetic Operator Configuration Dialog

In order to have some control on the values of the model parameters that the GA evolves, an upper and lower value can be set individually for each parameter using the ranges

dialog. This also helps avoid cases where the model solutions might generate an overflow (e.g. division by zero).

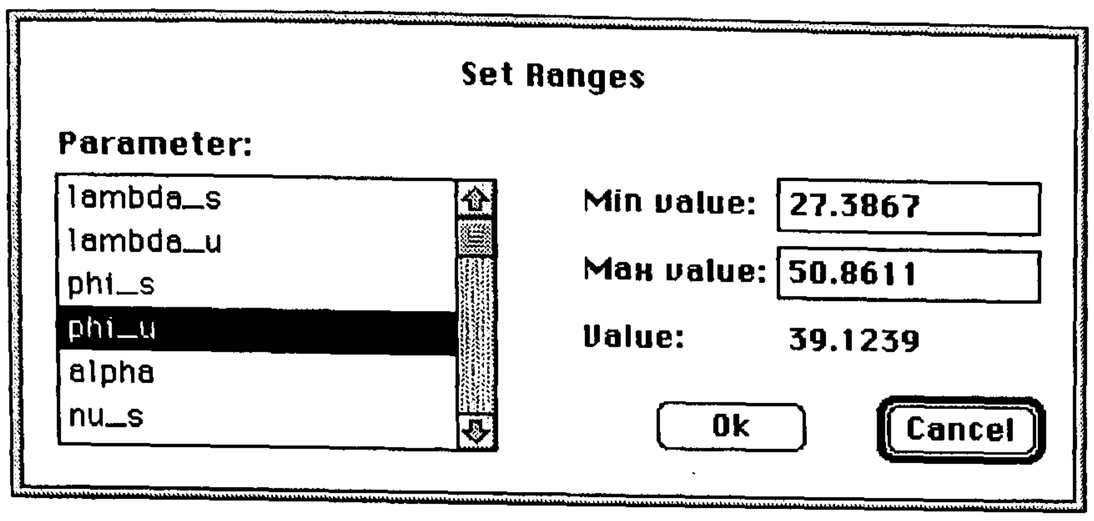


Figure 4.12 – Parameter Range Configuration Dialog

Finally, clicking on Optimise in the main configuration dialog starts the GA, which commences with the initialisation of the chromosomes in each pool. The dialog shown in Figure 4.13 appears while the GA is running, showing the current stage of the GA, and a value of the best and average fitness (calculated by calling the fitness evaluation function provided by GeoServer). At any time the GA may be stopped to examine the best evolved parameters so far.

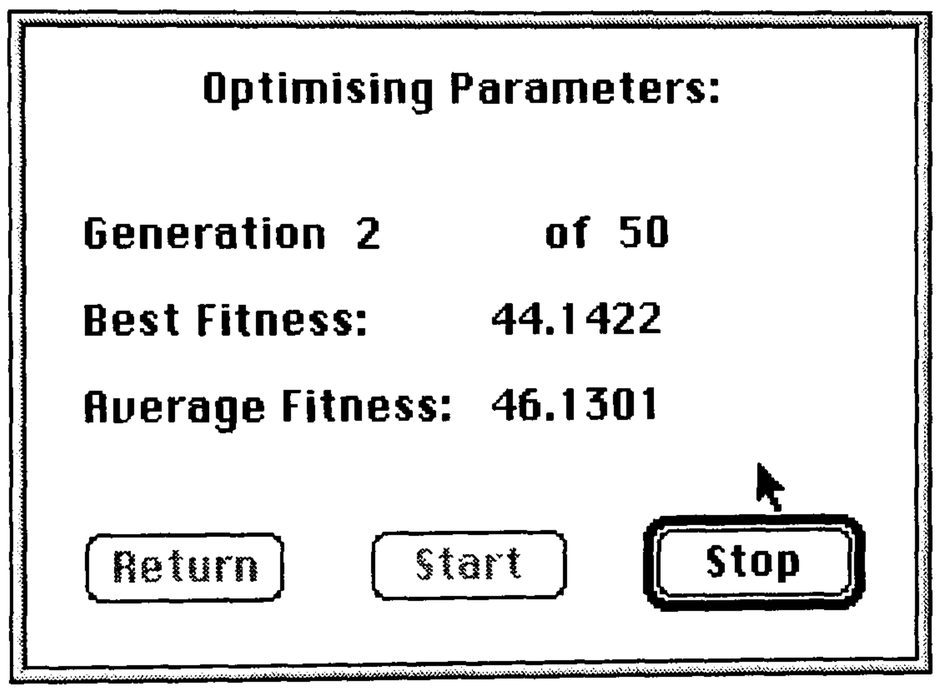


Figure 4.13 – GA Control and Feedback Dialog

The fitness evaluation function works by running the model from the first frame of the current dataset until the last available frame. This generates an equal number of frames which are then matched against the actual data to determine the goodness of fit.

The effectiveness of the GA will be examined in the next chapter, where it is used to evolve a set of parameters for a spatial model of high technology industry.

4.5 Decision Support

The decision process supported by GeoAnalyser typically (though not exclusively) consists of the following stages:

- **Definition of the spatial system**. Clarification of the extent and nature of the spatial system and it's variables.

- **Data acquisition**. Collation of all required spatiotemporal data.

- **Construction of the spatial model**. This will be used to project forwards in time. Suitable parameters reflecting policy measures are built into the model. Effects such as migration, commuting, population growth and decay, etc. are typically included.

- **Model calibration**. This will determine the set of model parameters that best describe the available spatiotemporal data.

- **Evaluation of the current system.** The calibrated spatial model is run forward over a number of timesteps starting from the most recent spatial dataset.

- **Evaluation of policy and environmental changes**. 'What-if' analysis is conducted by making changes directly to the spatial data and policy measures (reflecting, for example, local interventions that are planned, any changes that are expected in the system, etc.). The impact of such changes are observed by their effect on the model projections.

It should be noted that this decision process is often iterative. So, for example, if the model does not describe the data well during the calibration stage, revisions to the assumptions of the model and its mathematical form need to be made.

GeoAnalyser assists the decision process in several ways. Once the data has been obtained, the visualisation and analysis tools within GeoServer provide a means of exploring the dataset to develop an understanding of the system in terms of spatial patterns (through the thematic view), the time evolution of spatial variables (through the graph view) and spatiotemporal dynamics (using a movie of the thematic view).

After building a spatial model, these tools also assist in evaluating the model by visualising it's outputs. Finally the same tools may be used to evaluate the impact of policy measures (which are entered by altering the model parameters to reflect new circumstances and/or adjusting the value of any spatial variable to reflect local changes such as, for example, the creation of new jobs at a particular location).

GeoServer provides a means of developing and individually testing multiple models through it's plug-in architecture. The model itself is coded in C++ or C and an API ensures that it conforms to the plug-in interface.

Furthermore, the addition of a flexible, highly interactive user interface (greatly assisted by the MacOS GUI) promotes experimentation with the spatial model and investigation of the spatiotemporal dataset. No particular decision process is dictated and the user is free to experiment with the model and construct suitable views as required. This flexibility allows GeoAnalyser to be used to explore ill-structured and semi-structured spatial problems. The main limitation is the availability of suitable data and the ability of the modeller to develop a calibrated spatial model that adequately captures the dynamics of the spatial system. GeoAnalyser assists in that regard by offering facilities for testing different models within the plug-in architecture of GeoServer and the incorporation of a Genetic Algorithm in GeoClient for optimising model parameters.

Despite this rich set of features, GeoAnalyser has the following limitations:

- *Support for multi-variable views*. Only one variable can be plotted on both the thematic and graph views, making comparison between variables difficult (although views containing an arithmetic combination of variables can be constructed to get around this).

- *Linear data model*. The use of the linked list as the basis for the spatiotemporal data structure means that only a time trace can be examined at any time. A multiple time trace feature (essentially a branched linked list) would allow the comparison of different models or alternative policy measures. This could be achieved, to some extent, by running multiple copies of GeoAnalyser on the same set of machines to compare the outputs of different models.

- *Restriction to point datasets only*. Data related to entities having spatial extent must be converted into a point data set for inclusion into GeoAnalyser, thereby losing some of it's information.

- *Lack of a transport network*. Currently GeoAnalyser does not incorporate support for overlaying a transport network on the point dataset. Therefore, distances between points in the dataset must be measured 'as the crow flies'.

These limitations reflect the fact that GeoAnalyser is a research prototype and not a finished product. Additional features need to be added before it could be deployed as an end user tool. Nevertheless, as the next chapter shows, it can already be usefully applied for certain types of spatial analysis.

4.6 Application Areas

At the simplest level, GeoAnalyser can be used to explore any kind of point-based spatiotemporal data using the built in visualisation and analytical tools. It can also be employed for performing more complex decision support in a spatial planning context by using a spatial model to evaluate the impact of policy measures. Examples of both types of application are presented here.

The following tasks can be performed using GeoAnalyser:

- Investment planning in local, regional and national government (e.g. house building initiatives, business park development, etc.).
- Analysing the effect of policy measures on regional growth.
- Assessing the environmental impact of existing and future planned developments.
- Locational analysis and siting (e.g. locating a new retail complex, business park, leisure centre, etc. taking into account local population, existing amenities, and so on).
- Optimising the layout of new spatial developments (e.g. configuring a new housing development).

Due to its fundamental generality, GeoAnalyser can be used wherever a large amount of spatiotemporal data is involved in the decision making process. This leads to application in a very wide range of areas. The following are areas that might benefit from using such a tool:

- **Retail**: site selection, assessing and predicting store performance, sales territory planning, customer survey analysis
- **Finance**: competitive analysis, customer profiling
- **Insurance**: spatial risk assessment, market share studies, direct marketing, fraud detection
- **Utilities**: monitoring service delivery, analysing and projecting customer demand, fraud detection
- **Environment**: impact assessment, feasibility studies, resource inventory, disaster prediction, environmental monitoring

- **Government**: investment planning, forecasting, infrastructure analysis
- **Real Estate**: exploratory decision support for customers

4.7 Summary

This chapter has described the GeoAnalyser ISDSS in detail. This comprises a spatial model and genetic algorithm embedded in a spatiotemporal visualisation environment which uses direct co-operative coupling to link together the components of the homogeneous distributed system. The distributed design of the system takes advantage of the design autonomy, implementation autonomy and use of networked processing resources that can be realised using distributed computing methods. The data management mechanism, visualisation tools, analytical tools, and the methodology used to integrate the distributed components of the ISDSS were presented in depth.

GeoAnalyser supports many aspects of the spatial problem-solving decision process, through both the design of the user interface (which promotes the exploration of the problem through the facilities provided), the tools that it provides and features of the architecture which are designed specifically for spatiotemporal data (such as the linked list data structure which allows simple animation and graph displays).

The next chapter provides a working example of the use of GeoAnalyser for modelling the development of high technology industry.

Chapter 5

Application of GeoAnalyser to Industrial Modelling

This chapter provides a demonstration of the GeoAnalyser system for modelling the evolution of high technology industry in the South East of England. This is an industry that has been extensively studied in the literature but no attempts have yet been made to model it. From a definition of this industry, a detailed summary of modelling factors is provided, leading to the development of a spatial model. Next, the results of using the genetic algorithm to calibrate the model parameters are presented. The calibrated model is then tested for accuracy against unseen data. The model development and deployment process is used as the basis for discussing the design and implementation characteristics of GeoAnalyser.

5.1 Introduction

High technology industry (HTI) has received considerable attention in the urban studies literature in recent years. There is a perception that HTI development can lead to an agglomeration economy based on increasing returns (Faini 1984) where new firms entering an industry benefit overall from the presence of other firms. The apparent economic success of areas with a high concentration of this industry (Silicon Valley in California is the most quoted example of explosive high tech growth; Silicon Glen in Scotland and the apparent agglomeration of high technology firms around the M4 corridor in South East England are UK examples) have led to a belief among many planners that technology-oriented industries might be a panacea for reversing poor local economic conditions (MacDonald 1983). In the last decade there have been many efforts to attract computer and biotechnology firms to local areas in the hope of rejuvenating old and declining industrial areas (e.g. Wales). Such efforts are in vain if processes and conditions necessary for the development of HTI are not well understood.

Previous studies of high technology industry have been undertaken with a view to understanding the processes and mechanisms which underpin the growth of this dynamic industry. Studies have been conducted into the spatial agglomeration effects of HTI (Arthur 1989 presents a theoretical location-allocation process for new high tech companies locating in an increasing returns environment). The Analytic Hierarchy Process has also been used to model explicitly the static location choice decision process for HTI (Banai-Kashani 1990). This chapter takes a different approach by investigating

the possibility of modelling the development of HTI *empirically* by using GeoAnalyser to build and deploy a spatial interaction model. The first step is to determine the set of relevant modelling factors for the South East of England using the results of various studies. These factors are used as the basis for developing a nonlinear spatial interaction model, which is implemented as a plug-in for GeoAnalyser. The model is then calibrated using the built-in Genetic Algorithm and then tested against unseen data for validation.

The case study is used to test the modelling, decision support and spatiotemporal visualisation features of GeoAnalyser described in Chapter 4. The case study also assesses the operational characteristics of GeoAnalyser through it's application in a typical modelling task. This will provide useful insight for testing the viability of ISDSS in real world problem solving.

5.2 Overview of the Model Development Process

A simplified form of the process used to develop a spatial model is depicted in Figure 5.1. The preliminary stage is to define the scope of the model i.e. the entities and processes that need to be considered as part of the same model. The next two stages are related in that both rely mainly on data analysis and research to help determine the model variables and the internal processes that relate them. From these the mathematical model is assembled and (usually) programmed into a computer. The model is then calibrated against real world data in order to determine the values of the model parameters. Finally the calibrated model should be tested against unseen data as a validity check and then deployed.

For trivial models, most of the steps described will proceed in a linear sequence. However, for other types of model several of the steps may need to be repeated or re-evaluated. For example, if the calibration stage is not very successful then the model may need to be redesigned and modified.

Although this process does not show a lot of the detailed steps that are involved in building complex models (particularly those involving user requirements and data analysis), it does however indicate that the process of building, testing and deploying a spatial model is not trivial. Furthermore, this process is not unlike the software development process, which can be described as an iterative waterfall model (see Sommerville 1989 pp 6-12 for an introduction). Both types of development tend to undergo several iterations rather than progressing in sequence.

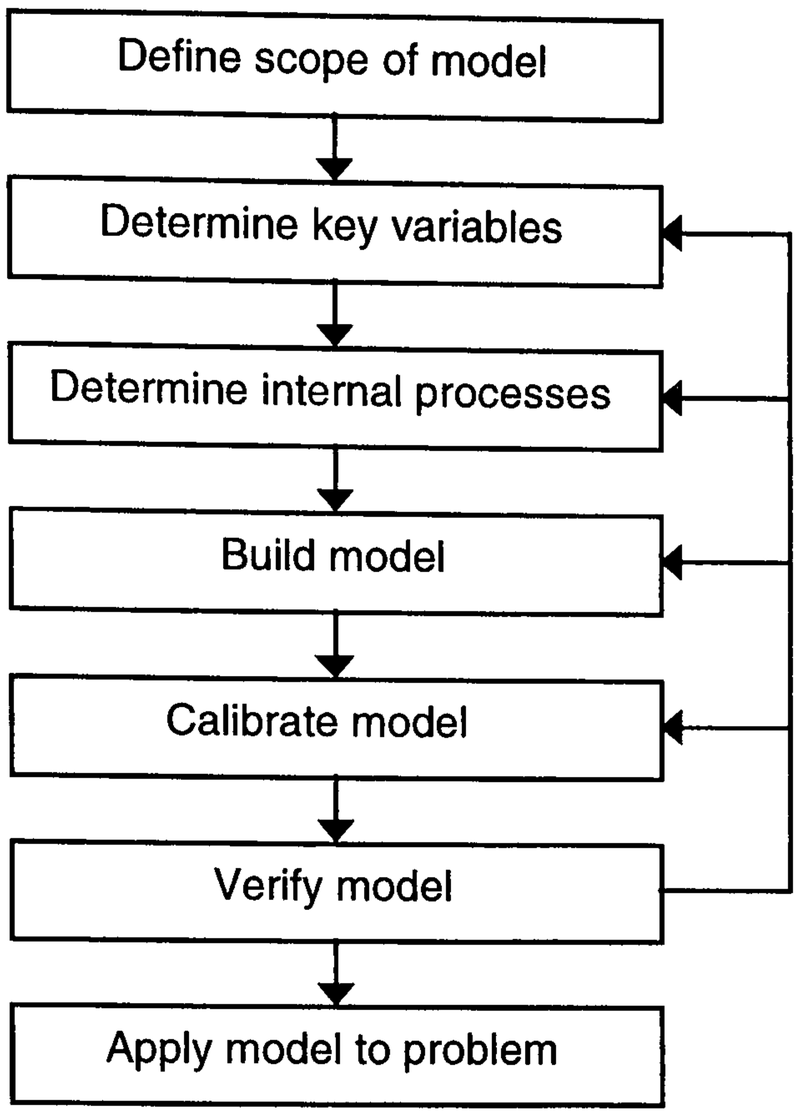


Figure 5.1 – The Model Development Process

The chapters that follow describe the steps involved in building a spatial model of HTI. This illustrates the application of GeoAnalyser to spatial model development.

5.3 Definition of High Technology Industry

Before beginning the task of building a model for HTI, it is important to clarify exactly what is meant by high technology industry. However, a clear definition of HTI is problematic (Worden 1986). Is an activity classified as high tech if it consumes *high technology inputs*, produces *high technology outputs* or uses a *high technology production process*, even though outputs may apparently be low tech (e.g. the use of hydroponics to grow vegetables)? Or should the level of R&D spend be used as the sole indicator instead? In fact, proposed definitions use a combination of such criteria to determine the inclusion of a particular industry as defined by the Standard Industrial Classification (SIC) Code.

The two most prevalent UK definitions are listed in Tables 5.1 and 5.2 (Hall, Breheny et al. 1987 and Butchart 1987 respectively). Hall's definition combines the level of R&D spend with the proportion of scientific and technical staff for the high tech manufacturing sector. Butchart's definition, on the other hand, differs mainly in the inclusion of high tech services.

1980 UK SIC Code	Description
AH 2570	Pharmaceutical products
AH 3302	Electronic data processing equipment
AH 3441	Telegraph and telephone apparatus and equipment
AH 3442	Electrical instruments and control systems
AH 3443	Radio and electronic capital goods
AH 3444	Components other than active ones, mainly for electronic equipment
AH 3453	Active components and electronic sub-assembly
AH 3454	Electronic consumer goods and other electronic equipment
AH 3640	Aerospace equipment manufacture and repairing
AH 7902	Telecommunications (excluding Post Office, broadcasting and local cable relay systems)

Table 5.1 – Hall's Definition of High Technology Industry

The choice between the definitions is a pragmatic one. The high tech manufacturing and services sectors have been shown to exhibit different locational tendencies and growth patterns (Begg and Cameron 1988). There is no correlation between high tech manufacturing and any other industry, including high tech services. Furthermore, in their study of HTI in 161 urban areas, Begg and Cameron found that the manufacturing sector shows no correlation with the size of the city. The services sector, on the other hand, does show a correlation.

Thus it is clear that the locational characteristics of the two sectors within HTI are very different. This has a direct impact on the modelling process. The approach adopted here is therefore to examine just the manufacturing sector, being the largest and hence the most important component of HTI, using Hall's definition to determine those industries that make up this component.

1980 UK SIC Code	Description
AH 2514	Synthetic resins & plastics materials
AH 2515	Synthetic rubber
AH 2570	Pharmaceutical products
AH 3301	Office machinery
AH 3302	Electronic data processing equipment
AH 3420	Basic electrical equipment
AH 3441	Telegraph & telephone apparatus & equipment
AH 3442	Electrical instruments & control systems
AH 3443	Radio & electronic capital goods
AH 3444	Components other than active ones, mainly for electronic equipment
AH 3453	Active components & electronic sub-assembly
AH 3640	Aerospace equipment manufacture & repairing
AH 3710	Measuring, checking & precision instruments & apparatus
AH 3720	Medical & surgical equipment & orthopaedic appliances
AH 3732	Optical precision instruments
AH 3733	Photographic & cinematographic equipment
AH 7902	Telecommunications (excluding Post Office, broadcasting & local cable relay systems)
AH 8394	Computing services
AH 9400	Research & development

Table 5.2 – Butchart's Definition of High Technology Industry

5.4 General Characteristics of High Technology Industry

Before attempting to model any system it is essential to understand the processes and factors that shape it's development. The following is a summary of the key factors responsible for and affecting the growth of HTI. Five main aspects were considered: *infrastructure*, *linkages*, *finance*, *labour* and *location*. These factors have been distilled from various research works examining the origin and development of HTI complexes in the USA, UK, France, Canada and Japan.

(i) Infrastructure:

- Government support for R&D (via tax cuts or the provision of public R&D centres - most importantly in defence) and major contributions to the demand side is essential at the pre-competitive stage (Saxenian 1983; Breheny and McQuaid 1987; Nishioka and Takeuchi 1987; Sayer and Morgan 1987).
- Prior industrialisation is not a requirement for encouraging new HTI growth. (Britton 1987).
- High tech products have large R&D costs (so global markets need to be addressed to recoup these costs) (Gillespie, Howells et al. 1987).
- Regions with a deep-rooted history in traditional industries have difficulty in adapting to the needs of HTI (Oakey 1984; Pottier 1987). However, those industries which can benefit greatly from new technologies are a suitable base on which to build HTI (e.g. machinery industries in Japan - car, watch and machine tool industries - gave rise to microelectronics firms through the demand for semiconductors in products and processes, particularly high-precision machine tools (Nishioka and Takeuchi 1987)).
- High quality accessible living environment with plenty of leisure activities (Feldman 1984; Taylor 1984; Breheny and McQuaid 1987; Pottier 1987).

(ii) Linkages:

- High tech agglomerations (high tech companies, public R&D labs, high tech users) form highly interconnected systems as a result of their many close internal interactions (Nishioka and Takeuchi 1987).
- High tech companies have mainly local input linkages (Oakey 1984).
- High tech companies have international output linkages - only a small number are in the local agglomeration (Oakey 1984).
- Close supplier-customer relations (caused by rapid development and short product lifetimes) gives rise to high spatial cost of information dissemination (i.e. face-to-face relations are a subtle advantage for innovation) and therefore forces agglomeration (Saxenian 1983; Oakey 1984; Henderson and Scott 1987).
- Short product lifetimes result in a knowledge/skill barrier to new entrants (Gillespie, Howells et al. 1987).
- Downstream users in the local area (the custom-built nature of many high tech products favours those companies located close to users in terms of forming correct specifications of the product and learning from the user in order to improve the product further) (Breheny and McQuaid 1987; Henderson and Scott 1987).
- Links with HEI's are vital (Taylor 1984).
- HEI links with high tech companies appear to be unconstrained by location (Gillespie, Howells et al. 1987).

(iii) Finance:

- Specialised banking/venture capital services to provide start-up capital (Saxenian 1983; Taylor 1984; Henderson and Scott 1987).
- Investment capital for subsequent expansion is overwhelmingly drawn from internal profits (Oakey 1984).
- Venture capital tends to form a local feedback system in that profits are generally ploughed back into the same area for reinvestment (Oakey 1984).

(iv) Labour:

- No unionisation in high tech companies (Gillespie, Howells et al. 1987; Henderson and Scott 1987).
- High staff turnover rates (Saxenian 1983; Henderson and Scott 1987).
- Large proportion of skilled technical/professional workers (40%) (Saxenian 1983; Feldman 1984; Taylor 1984; Breheny and McQuaid 1987; Henderson and Scott 1987).
- Progressive employment practices (mainly American and Japanese firms) provide a good working environment, which is attractive for technical/professional staff (Gillespie, Howells et al. 1987). This is also important for innovation (Oakey 1984).
- Significant proportion of the workforce is unskilled/semi-skilled (mainly for assembly-type work) (Saxenian 1983; Markusen 1984; Gillespie, Howells et al. 1987).
- Low requirement for skilled manual workers (due to automated production processes) (Gillespie, Howells et al. 1987).
- Level of technical skill increases with the size of the agglomeration (Pottier 1987).
- Labour shortages can form a significant obstacle to growth (Oakey 1984).
- New firms have high levels of office-based work (R&D, administration, marketing) than established firms, which have higher levels of production (Breheny and McQuaid 1987).

(v) Location:

- HTI tends to locate close to large cities (Breheny and McQuaid 1987).
- Good telecommunications are essential (Pottier 1987).
- Good road transport system in general (Feldman 1984; Taylor 1984; Breheny and McQuaid 1987).
- Access to an international airport (Feldman 1984; Taylor 1984; Breheny and McQuaid 1987; Pottier 1987).
- Availability of good business premises (Breheny and McQuaid 1987).
- There are significant locational differences among the various sectors constituting HTI, possibly attributable to differing labour requirements (Markusen 1984;

Breheny and McQuaid 1987; Pottier 1987). An example is biotechnology (Feldman 1984).

- Mature HTI tends to decentralise some activities, such as mass production centres, away from cities (Breheny and McQuaid 1987; Nishioka and Takeuchi 1987).

In addition to these factors, two further observations can be made regarding the nature of high tech growth. First, it is clear that HTI cannot necessarily rejuvenate stagnant areas based on traditional industries as it's growth requirements are not usually met by such locations (Breheny and McQuaid 1987; Gillespie, Howells et al. 1987). Secondly, several studies point out that HTI growth is characterised by the creation of spin-out companies, particularly from large indigenous companies (Saxenian 1983; Breheny and McQuaid 1987; Britton 1987; Henderson and Scott 1987).

5.5 High Technology Development in South East England

The South East of England is popularly reported to have the highest concentration of HTI in the UK, with the greatest proliferation supposedly being the "M4 Corridor", the so-called stretch of the M4 motorway between London and Bristol. However, research into the current demographics of HTI has revealed that this perception is only partly true (Breheny and McQuaid 1987; Hall, Breheny et al. 1987).

In their extensive study of the South East, Hall et al discover that in fact the corridor is more like a broad crescent covering Berkshire, Hampshire and Hertfordshire. The history of the development of the "Western Crescent" as it is termed by Hall et al sheds useful insight for the development of the spatial model and therefore is summarised below.

Although it is unclear exactly why, in the 1920s and 1930s West London became the centre of the HTI of the time in electrical goods production (gramophones, radios and televisions), accounting for 46% of the national total in this industry. Due to high location costs in London, many of the companies in this sector and also in aerospace relocated out until around the 1950s to Berkshire, Hampshire and Hertfordshire. This was influenced by the location of the Defence Research Establishments (which were already concentrated in the counties surrounding West and South West London; this concentration was increased through the development of a military base at Aldershot and the Royal Aircraft Establishment at Farnborough) as the impact of defence procurement for the Second World War became felt in the electrical and aerospace industries. The need for face-to-face meetings for the development of new products and technologies meant that proximity to the client was crucial. The need to remain in the South East was already clear for these companies as it provided a ready supply of graduates and local markets for goods produced. Furthermore, in the 1980s, defence work was an important mainstay for

these companies as the rest of the economy experienced a slump while defence spending increased.

In addition to these factors, planning authorities had an important part to play. The designation of Heathrow as the primary air terminal at the end of WWII and the development of the M4 motorway west of London in the 1970s increased the attractiveness of this area. The Greater London Plan (Abercrombie 1944) also concentrated investment in areas such as Slough with the creation of it's large industrial estate and the development of a new town (Bracknell) in Berkshire. Indirect planning was also significant as electrical companies found it much easier than other companies to obtain the requisite Industrial Development Certificates in the South East and in new towns neighbouring London.

Breheny and McQuaid conjecture that the planning restraints imposed on new developments have actually favoured HTI as these have helped to maintain the amiable semi-rural environment that is preferred by a great many high tech firms. They also conclude that the efforts of the public sector as described above have been largely accidental rather than part of a concerted effort to encourage the development of HTI in the South East.

Breheny and McQuaid's survey of 44 Berkshire electronics firms reinforced the view that spin-outs play an important part in the growth of HTI. They discovered that most the 24 single plants (i.e. those that are not part of larger UK or overseas company) had been spun-out from electrical companies in West London and that proximity to the home of the founder, who commuted in from Berkshire, was among the most important locational factors. Interestingly, they found that the presence of a nearby university did not rank highly.

These findings echo the factors that were described in the previous section but reinforce the importance that the defence industry had for the development of HTI as a whole in the South East.

5.6 Data Requirements for Modelling High Technology Industry

The modelling factors described lead naturally to the next stage of the modelling process. This is to determine the entities that are to be considered as part of the system to be modelled. The following is a list of entities that would ideally be included:

1 Concentration of hi tech labour by labour category (technical/professional, skilled manual and unskilled manual)
2 Concentration of hi tech companies

3 Level of R&D spend by hi tech companies
4 Concentration of Higher Education Institutes
5 Concentration of Government-funded R&D establishments
6 Average level of house prices
7 Average level of industrialisation
8 Availability of business premises
9 Availability of Government startup capital
10 Availability of finance and venture capital

Each entity needs to be disaggregated spatially according to subregion† and temporally over the last 20 years or so with annual increments. This level of disaggregation is problematic for several reasons. First, although current values of these entities are relatively easy to obtain, it has been found that suitable historical data is either not available or incomplete. Secondly, in those cases where data is available through, for example, annual census data (e.g. Census of Employment), government guidelines strictly limit the level of disaggregation that is permitted for public disclosure (this is to provide some level of protection for individual companies and organisations). This covers spatial disaggregation and also sectoral breakdown. So, for example, very detailed data on the pharmaceutical products industry (SIC code AH 2570) are not available at the ward level. For this reason, census data is not available publicly at this level of disaggregation.

A further difficulty with data is that Census of Employment extracts are charged for by the UK Central Statistical Office. The full dataset, covering annual data over 20 years and spatially disaggregated to the county level (the highest permitted level), would cost several thousand pounds. This was unfortunately beyond the funds available.

Due to these difficulties a compromise was reached. Ten years of annual data (covering 1981 to 1990) for the high technology sector were obtained from the Central Statistical Office. These were spatially disaggregated for the counties of South East England for the following entities:

- number of companies
- number of administrative, technical and clerical staff
- number of operatives (i.e. production workers)

The data were obtained in aggregate form for the industries making up the high tech

---

† Subregion refers to small extents within a region (such as the South-East of England). Typically, there will be about fifty subregions in a region. A subregion should ideally be smaller than a county but larger than a parish, which is the smallest geographic area covered by any Census data.

sector due to the confidentiality issue.

Data on government and defence research establishments (GREs) are extremely difficult to obtain, due to security measures imposed since World War II. Hence, data that can be used to gauge the relative size of the GREs are not publicly available. However, Hall et al list the GREs in the South East (totalling 22 for the whole region). This was used to determine the number of GREs for each county.

A similar list of higher education institutes (HEIs) was obtained from the Department of Education (Department of Education 1993). Unfortunately, historical data on staff and student numbers was not available and so a count of the number of HEIs was used instead.

In order to compare the level of high tech manufacturing compared to other industries, data for the number of employees in the manufacturing sector as a whole were also obtained from the Census of Employment.

The remaining historical data were either not available or were prohibitively expensive to purchase. Therefore the following data items will be used for the modelling process:

- number of high tech companies;
- number of administrative, technical and clerical staff employed by high tech companies;
- number of operatives employed by high tech companies;
- total level of employment in the manufacturing sector;
- number of HEIs;
- number of GREs.

Clearly this imposes a severe constraint on the modelling that can be performed. However, it should be sufficient for the purpose of validating the design of the GeoAnalyser system.

5.7 Analysis of South East England (1982-1986)

Having determined the variables to be used for the basis of the model, the relevant data was then obtained. Only the years from 1982 to 1991 inclusive were obtained (as per the cost restrictions mentioned earlier). The strategy that was adopted was to use the 1982-1986 data for building the model and the remaining half to test the nonlinear model.

The next stage was to examine the data obtained for the study area. This was to build further understanding of the study area as a necessary part of the model development process.

An analysis of the data revealed the following characteristics for the data sets over the period from 1982 to 1986 inclusive: (i) high growth in Oxfordshire hi-tech employment (170%) and companies (230%); (ii) large volume increase in Hertfordshire skilled labour of 11% compared with a slight fall in unskilled labour of 1%; (iii) high growth in Bedfordshire unskilled labour (46%) but a small decrease in skilled labour (6.5%); (iv) large rise in total manufacturing employment of 28% in Buckinghamshire. Each of these apparent anomalies were subjected to further analysis.

(i) Oxfordshire

The large rise in Oxfordshire companies is most likely due to new company formation. This is mainly attributed to the creation of spin-outs (Bradstock 1994; Stott 1994). For example in cryogenics, one firm, which began by creating powerful magnets for research at the University, generated 6 spin-outs over 1982-86 due to demand from overseas buyers (Lawton Smith 1991). Cultural changes during this time would also have played a part. The most likely driving force for the large growth must surely be Oxford's academic excellence which found the climate of the early 1980s so favourable. The opening up of European and international markets may have also played a part. The local support infrastructure provided by the University and local government bodies would have helped to ensure that the growth was long term, although the Oxford Science Park did not open until 1989.

(ii) Hertfordshire

Hertfordshire has approximately the same number of skilled high tech workers as Greater London but a much smaller number of unskilled workers. This is attributed to the high growth of pharmaceuticals, telecommunications and finance (mostly relocations from London). Labour intensive industry, on the other hand, has tended to relocate out of Hertfordshire. According to the Census of Employment half of the total workforce is employed in large companies, which make up only 5% of the total number of companies in the county.

During the 80s, Hertfordshire had one of the highest rates of new company formation in the country, on average this was 1.3% higher than the national figure over 1982-86 (based on VAT Registration Data). This was probably due to its proximity to London and the strong economic links it has with the capital. The major growth sectors were financial services (98% rise), instrument engineering (78% rise) and chemicals (21% rise for 1985-90) while manufacturing industry experienced a steep decline of 23%.

According to an internal report of the Hertfordshire County Council, there are several reasons for the county's growth (Booker and Pryor 1994). Its proximity to the M25,

which runs through the county and was completed in 1986 has helped to improve road links. Hertfordshire was the site of many garden cities and post-war new towns (Hemel Hempstead, Welwyn Garden City, Stevenage, Letchworth and Hatfield). It has many large employers in aerospace/defence, electronics, pharmaceuticals and computers. The main HEI is the University of Hertfordshire.

(iii) Bedfordshire

Bedfordshire has the highest proportion of manufacturing industry in the South East outside London. Changes in the B1 use classes of land would have had a positive effect on light manufacturing, permeating as a rise in unskilled labour only. Coupled with the fact that numbers of companies have also risen, the increase would most likely be reflected in the level of unskilled labour, given the high representation of manufacturing in the county.

(iv) Buckinghamshire

This was mainly due to the growth of the Milton Keynes new town from the mid 70s onwards; incentives were provided to facilitate growth during this time (Liddiard 1994; Yaar 1994). Manufacturing industry was the main employer (particularly electronics companies and large multinationals).

The picture that emerges from these investigations is that local factors have a part to play in the relative growths of high tech industry. However, these factors are outside the scope of what can be reasonably modelled given the lack of sufficiently detailed data. In any case, no model can capture every aspect of a system and there will always be factors that need to be considered external to the system under analysis.

5.8 Nonlinear Growth Model

According to Arthur (Arthur 1989):

"If locational increasing returns are present in the industry location problem, static equilibrium analysis is not enough. Settlement patterns are path-dependent; we are forced to follow the locational structure as it forms, and this requires dynamic analysis."

What is needed is a methodology that can model both settlement patterns for new entrants and also the evolution of existing firms. The approach adopted here to model HTI is based on evolutionary models as applied to central places (Allen and Sanglier 1979; see Engelen 1988 for a useful introduction). These ideas are strongly related to theories of complex systems in the physical and mathematical sciences (Prigogine and Stengers 1984), also known popularly as Chaos Theory. This approach models the urban system as

a set of discrete locations using a set of model equations that evolve the system through a series of bifurcations – points of instability during which fundamental structural changes and self-organisation can take place. This seems to be the most fruitful way to proceed in recognition of the dynamic, apparently self-organising nature of HTI.

The approach taken was to attempt to model the manufacturing component of HTI as a separate entity. This was justified by the fact that this sector bears no direct correlation with any other industry (Begg and Cameron 1988) and can therefore be modelled separately.

The spatial model of high technology industry is based on nonlinear logistic growth equations that express how the following entities interact:

- Skilled and unskilled high tech labour at each location (E_i^S and E_i^U respectively)
- Number of high tech companies (C_i)

The nature of the interaction is such that if there is an excess of employment at a location then there will be a positive influence on the number companies at that location that can make use of this excess labour. The converse is also true: if the number of companies exceeds the labour capacity of the location then some of these companies will either relocate to other locations in order to grow or go bust (through competitive pressure).

This simple picture is complicated somewhat by the inclusion of migration effects that take into account changes in the number of companies according to the relative attractivity of a location compared with other locations in the study area. Positive attractivity factors include the relative number of higher education institutes (HEIs), the number of government research establishments (GREs) and the skilled labour potential. The distance between two location is a negative attractivity factor (i.e. distant locations are less attractive than those that are nearby).

The change in companies can then be modelled as follows:

rate of change = local growth + migration in + migration out

$$\frac{dC_i}{dt} = \alpha C_i\left(\sum_{k=S,U} \nu_k E_i^k - C_i\right) + MC_i^{in} - MC_i^{out}$$

where the local growth is modelled as the logistic term. The parameter ν_k weights the growth potential of each type of high tech employment.

The term for the migration of companies *into* a location, MC_i^{in}, is given by the sum over all locations j excluding i of the number of companies that migrate into i from j according to the relative attractivity of i compared to all other available locations i'. This is described by the following term:

$$MC_i^{in} = \gamma \sum_{j \neq i} C_j \frac{A_{ij}}{\sum_{i' \neq j} A_{i'j}}$$

where γ modulates the migration effect and A_{ij} is the relative attractivity of location i compared with location j.

The term for companies migrating away from a location, MC_i^{out}, is given by a similar term with modified indices. The migrating quantity, C_i, can now be taken outside the summation as it is now common to all the summation terms.

$$MC_i^{out} = \gamma C_i \sum_{j \neq i} \frac{A_{ji}}{\sum_{j' \neq i} A_{ij'}}$$

The attractivity term A_{ij} is described as an exponential growth curve based on the sum of the attractivities of each of the factors described above:

$$A_{ij} = e^{\varepsilon_h (h_i - h_j) + \varepsilon_g (g_i - g_j) + \varepsilon_E ((\phi_s C_i - E_i^S) - (\phi_s C_j - E_j^s)) - \varepsilon_d D_{ij}}$$

where,

h_i is the number of HEIs at location i;

g_i is the number of GREs at location i;

D_{ij} is the distance between locations i and j;

ϕ_s measures the potential for skilled labour;

and the ε parameters measure the relative strength of each attractivity.

The remaining components of the model are concerned with the growth of skilled and unskilled labour. As the driving forces for growth are encapsulated within the differential equations for the number of companies, the growth of labour can be described as the following simple logistic equation:

$$\frac{dE_i^{k=S,U}}{dt} = \lambda_k E_i^k \left(\phi_k C_i - E_i^k \right)$$

There are a total of 12 parameters that need to be calibrated using GeoAnalyser's model optimisation tool and the sample dataset.

5.9 Model Calibration

The process of calibrating the spatial model equates to that of finding the best set of parameters such that the model dynamics match as closely as possible the dynamics of the sample dataset (i.e. the calibration data). Although it would be highly desirable to be able to fully automate this process, which is essentially one of searching the 12 dimensional space defined by the model parameters, a number of difficulties exist that prevent this from being achieved. First and foremost, due to the complex interactions present in the model (resulting from the need to model migration), many (probably the vast majority) of the possible parameter values results in divergent solutions. Such divergence makes it very difficult to be able to compare the relative fitness of two sets of parameters in the genetic algorithm. Secondly, as the model is computationally quite intensive, it is not practically feasible to search the entire parameter space to find the best solution using a personal computer.

The only practical solution is therefore to limit the parameter search in some way so that reasonable solutions can be evaluated and hopefully improved upon to get the best solution for that partition of the parameter space. A two stage methodology was adopted in order to achieve this. The second stage is described more fully in Appendix A.

First, a set of approximate parameters was determined using a combination of calculation and trial and error. The calculations were performed using a spreadsheet and the parameters were tested using GeoAnalyser, with the criterion being that the results should approximate the calibration dataset within an order of magnitude. This was a fairly straightforward process and resulted in the set of parameters shown in Table 5.3.

Secondly, the approximate parameters were used to construct a prototype individual which was then used to seed the gene pool. For the prototype individual, each chromosome was assigned a range that limits the values that the parameter can take. This range effectively partitions the parameter space so that reasonable solutions can be evolved. After each generation the range of each parameter was re-evaluated using the minimum and maximum values of that parameter for each individual in the pool and allowing an additional 30% margin for parameter migration.

The following error function was used to determine the fitness of an individual based on the discrepancy between the simulation results and the real (i.e. actual) data:

$$\varepsilon_{ik} = \left(\frac{v_{ik}^{sim} - v_{ik}^{real}}{v_{ik}^{real}} \right)^2$$

Parameter	Value
λ_s	0.00001
λ_u	0.00001
ϕ_s	43.4
ϕ_u	39.1
α	0.0001
ν_s	0.012
ν_u	0.012
γ	0.0001
ε_h	0.0001
ε_g	0.0001
ε_E	0.0001
ε_d	0.0001

Table 5.3 – Initial Parameter Values

where k and i denote the particular spatial variable and spatial region respectively, and v is the value of the spatial variable k at region i.

The GA was run several times using different numbers of individuals and pool sizes. The GA was halted when no further improvements to the best individual were discernible. Figures B.1 to B.4 (see Appendix B) depict the optimisation process under various different configurations. Table 5.4 details the results and configurations used. In each case, the following genetic operators and rate parameters were used (these are already included in the QGAME package):

- random initialisation
- roulette wheel selection
- basic mutation (mutation rate = 0.03)
- chromosomal crossover (crossover rate = 0.6)
- best-to-worst migration (migration rate = 0.3)

	Figure B.1	Figure B.2	Figure B.3	Figure B.4
Best Fitness	10.43	10.39	10.25	10.24
Pool size	30	50	50	50
No. of pools	1	3	5	5
λ_s	8.00E-12	8.70E-08	3.511E-08	6.09E-08
λ_u	2.00E-12	2.00E-08	8.06E-10	1.40E-08
ϕ_s	11332	41.71	43.56	9.117
ϕ_u	66818	36.38	41.42	4.889
α	1.77E-08	0.000184	0.000194	0.000196
ν_s	30.770	0.03455	0.03316	0.04319
ν_u	3.890	0.01324	0.02826	0.01310
γ	2.04E-10	2.2E-05	2.556E-7	3.01E-08
ε_h	9.90E-11	3.62E-07	9.5E--05	4.25E-08
ε_g	3.58E-10	4.49E-07	1.3E-05	5.28E-08
ε_E	3.10E-11	2.32E-07	9.1E-05	4.58E-09
ε_d	2.00E-11	0.000085	5.4E-05	7.26E-10

Table 5.4 – Summary of Model Calibration Results

Several interesting observations can be made about the results. First, it is clear that the GA has helped to calibrate the model up to a point – this is illustrated in the graphs in Figures B.1 to B.4, showing clearly the improvement in model fitness over the generations. In most of the runs, the GA achieved a best solution in a relatively small number of generations (around 50). The graphs of the average fitness also show that the population gradually became divergent even though the parameters were constrained within a bound. However, after a few hundred generations (as shown in Figure B.1) the average solutions suddenly settled down before again gradually starting to fluctuate. This adds weight to the suggestion that optimisation of a spatial interaction model is difficult. In spite of this, the GA still managed to deliver reasonable results.

Further examination of the results reveals that best solutions that were evolved in each run of the GA are appreciably different from each other. Looking back at the model

parameters suggests that different sets of parameters could indeed generate similar results (due to the existence of a certain degree of redundancy in the way the parameters have been defined). This could also be caused by parameters becoming so small as to have very little effect on the overall solution (such as the parameters, which show the greatest variance across the different calibration runs). A further possibility is that these different sets of parameters are due to the presence of local minima in the fitness surface. This would also explain perhaps why no better solutions were evolved.

The best overall individual achieved a fitness of 10.24. Given that there are 13 regions with 3 calculated variables for each region (with the other variables remaining fixed) and four sets of data that were compared (giving a total of 156 distinct comparisons), this translates into an average error of 26% per variable. While in many situations this would be considered poor, considering the type of model that is being optimised here, this is not a bad result.

The next step is to test how well the predictions generated by the three sets of parameters compare with unseen real data.

5.10 Prediction Testing

To test the prediction capabilities of the evolved solutions, GeoAnalyser was initialised with each set of parameters in turn with 1987 spatial data (from the second half of the dataset that was used for the calibration). The model was then projected forward in time using a time step of 0.01 years in order to generate projected data for the years 1988, 1989, 1990 and 1991. These projected data sets were then compared with the actual data for those from the original dataset. The real and projected data were compared using simply the average error. The results are summarised in Table 5.5.

Fitness	Average Error
10.43	17%
10.39	19%
10.25	20%
10.24	20%

Table 5.5 – Model Prediction Summary

These results suggest that the evolved parameters are reasonable, generating at worst a 20% error for prediction over 4 years.

In order to provide a comparison, predictions were also made using linear regression with the aid of a spreadsheet program (Microsoft Excel). The five spatial data points covering the period 1982-1986 were used as the basis to calculate the values of the variables in 1988, 1989, 1990 and 1991. The average error for the regression over this period was calculated to be 30%.

Clearly the model developed using GeoAnalyser surpasses the results obtained using regression by a considerable margin. However, one could argue, considering the amount of calculation and work involved, that perhaps the system should be capable of an even better result. This is perhaps quite true and ordinarily a modeller would probably try out several slightly different models, calibrating each one and then evaluating the calibrated model's accuracy for prediction. However the result obtained using just the one iteration is sufficient for the purpose of validating the design of the GeoAnalyser system and demonstrating its usefulness as a tool for model calibration.

5.11 Performance Assessment

The performance of GeoAnalyser was assessed under two different hardware configurations. The first used a serial Appletalk cable to directly link two computers together. In the second arrangement both computers were connected via a departmental Ethernet network running at 10 Mbps under normal load conditions (i.e. during office hours). Unfortunately various constraints prevented the use of the same client and server computers under the two network configurations. Full descriptions of the client and server machines used are shown in Table 5.6.

Also shown in the table are the results of the two sets of tests that were carried out to measure the performance of the hardware:

- *Relative hardware speed.* A standard benchmarking application called *Speedometer* was used to assess the relative speed of the machines. This provides separate measurements for CPU, graphics speed, hard disk speed, mathematical operations as well as a overall performance rating with higher numbers indicating greater speed.

- *Average network speed.* Measurements were made of the length of time taken to transfer files of various sizes between the two machines in order to provide a comparison of the peak speed of the two network configurations.

	Computer			
	Client 1	**Server 1**	**Client 2**	**Server 2**
Model	Powerbook 160	Power Macintosh 6100	Quadra 840 AV	Power Macintosh G3
Memory/ Hard disk	8MB/80MB	24MB/500MB	16MB/500MB	32MB/2GB
Processor type and speed	33 MHz 68030 + floating point unit	66 MHz PowerPC 601 (integral floating point unit)	40 MHz 68040 (integral floating point unit)	233MHz PowerPC 750 (integral floating point unit)
CPU rating	0.52	3.45	1.41	18.40
Graphics rating	0.39	2.04	1.36	n/a
Hard disk rating	0.30	1.50	1.62	4.23
Math rating	4.36	117.41	24.15	649.86
Overall rating	0.45	2.52	1.58	14.45
Network speed	16.49 kbps		162.34 kbps	

Table 5.6 – Description of Test Hardware

A second set of tests were conducted to measure the performance of GeoAnalyser on the two hardware configurations. For the GeoAnalyser tests, all 10 temporal data points associated with the high technology sector in South East England were used (covering years 1982 to 1991 inclusive). The following tests were carried out:

- *Display*. To assess the speed of the user interface where operations involve interaction with the server, four different tests were undertaken. In each test, the Task Manager was used to animate through all 10 temporal data points and the time taken recorded using a stopwatch. In three variations of the test, one additional display window was also open at the same time, making a total of four distinct tests: (i) Task Manager alone; (ii) Task Manager and Region view; (iii) Task Manager and Graph view; and (iv) Task Manager and Table view.

- *Model execution*. To assess the execution speed of the spatial model, the model was run for 100 timesteps by pressing the *Run For* button on the Task Manager. Two versions of the test were run: (i) where all 100 intermediate frames were generated (*Save Memory* feature turned off) and (ii) where only the last frame was generated (*Save Memory* feature turned on). It will be noted that the server

provides separate Apple Event methods for these two situations and so quite different results were expected.

- *Model calibration*. The speed of the genetic algorithm was also assessed by measuring the time required by the system for the following stages of the optimisation process: (i) initialisation of individual population; (ii) first generation and (iii) second generation. The reason for including the second generation was that it was found that the first generation usually took far less time to compute than later generations, as the fitness of all members had already been calculated during the initialisation stage. In all cases, default settings were used for the GA, operators and operator parameters. The timestep used was 0.1 years in all cases.

In order to determine directly the overhead or benefits of the distributed architecture of GeoAnalyser, a non-distributed version of the application was also constructed. Identical code was used for the bulk of the application. The same set of tests were performed using the non-distributed version on both client machines.

All response times were measured using a stopwatch as before. Each test was run three times in order to even out the effects of network traffic and obtain more accurate measurements. The results of these tests are summarised in Table 5.7.

Test		Hardware Configuration			
		Client 1 - Server 1 (s)	**Client 1 only (s)**	**Client 2 - Server 2 (s)**	**Client 2 only (s)**
Display:	**(i)**	2.16	0.78	1.89	0.87
	(ii)	11.36	1.60	9.96	1.86
	(iii)	32.66	1.17	31.26	1.30
	(iv)	3.15	0.90	2.53	0.85
Model Execution:	**(i)**	30.61	192	5.08	48.25
	(ii)	7.66	181	1.97	44.26
Model calibration	**(i)**	42.00	3435	10.85	875
	(ii)	7.30	518	2.13	217
	(iii)	19.74	1461	4.23	375

Table 5.7 – GeoAnalyser Performance Results

The results show that, for both hardware configurations, the distributed version of GeoAnalyser performs significantly slower on all display tests than the non-distributed version. This is particularly striking in tests (ii) and (iii), where the response times are very similar for both hardware configurations. This implies that some hard limit has been reached irrespective of machine speed and network bandwidth. The results suggest that just 3-10 Apple Events per second can be exchanged. This clearly puts a severe constraint on the use of Apple Event technology for user interface tasks. However, all of the display operations basically use static data, which were repeatedly retrieved from the server. This is particularly noticeable for the graph view, as each redraw of the graph requires ten messages to be passed to the server. This aspect can obviously be optimised by using a client side data cache that would be updated from the server only when data were modified. This should completely alleviate the performance penalty incurred by unnecessary server communication.

A further observation is that other forms of interapplication communication do not generally suffer from the same performance bottleneck. For example, Web server software running on a relatively slow machine can make tens if not hundreds of socket connections per second.

On the other hand, the results for the model-related tests show a *huge performance gain* when compared with the non-distributed version. These tests run as much as 95 times faster on the distributed version of GeoAnalyser. This convincingly demonstrates how the computational power of the server machine can be extended successfully to the client machine.

It should also be noted that the two hardware configurations exhibit very similar performance, taking into account the relative speed of the machines. This was to be expected as the relative performance advantages of the server machine over the client is similar for both configurations.

5.12 Design Assessment

Preceding sections of this chapter describe the use of GeoAnalyser for the task of developing and testing a spatial model of high technology industry. It is at this point worthwhile making some preliminary observations regarding the design aspects of the system. Clearly detailed assessment of these aspects requires external evaluation. Unfortunately this was not possible as the original user, KPMG Peat Marwick, underwent organisational changes that prevented it from becoming a pilot user group.

The following general observations were made:

- *Flexibility of user interface.* The use of a graphical interface with separate windows for each display type affords a large degree of flexibility as users can decide for themselves those aspects of the dataset that they are interested in and arrange the visual elements accordingly. The variety of visual components was found to be useful and the animation features offered a partial solution to the difficult task of visualising spatiotemporal data on a computer screen.

- *Correctness.* This is obviously a critical aspect of the system. Careful checking was required during the model development stage to ensure that the system generated results that were consistent with the model being used.

- *Robustness.* At the time of writing, it was found that the system would occasionally crash. Although this can be attributed to memory allocation issues, it is worth recognising that known problems exist with the class library that was used to build the user interface.

- *Decision support.* The flexibility of the exploratory user interface was found to be very constructive. This does however lead to a danger that inexperienced users will not know the best way of using the system for a particular task. On-line tutorial and help facilities would therefore be required. Finally the lack of a facility to store concurrently multiple sets of model parameters as well as multiple model projections at the same time made it difficult to directly compare different sets of parameters and results obtained during model deployment (e.g. what-if analysis).

- *Model development.* The visualisation and data analysis support within the system was found to be generally useful but not complete enough to allow complete analysis within the system. Thus in practice a spreadsheet package was used in conjunction with GeoAnalyser, for example, to explore the behaviour of functional forms to assist with development of the model equations and to compare the results of different model runs or types of model. The relatively primitive tools for importing and exporting data sets were a limitation here.

- *Model calibration.* The model calibration functions were found to work well and effectively, particularly in the distributed version of the application.

- *Model deployment.* This aspect of the system was not explored in the tests that were run as it was felt that this can be done effectively only with a real user group.

Clearly the current system has many benefits for the various stages involved with spatial analysis. It is also evident that the system has limitations, many of which may be attributed to the fact that it is still at the research prototype stage.

5.13 Summary

This chapter has described the application of GeoAnalyser system to the real world task of building a model of high technology industry based on data for South East England from 1982 and 1986. The system was used throughout this process, beginning with exploration of the spatiotemporal data set, calibration of the model (using the genetic algorithm built into GeoAnalyser) and model testing (against data for the same region for the period 1987 to 1991). The accuracy of the evolved parameters was found to be superior to regression while offering the additional benefits of what-if analysis.

The performance of GeoAnalyser was evaluated using various hardware and network configurations. It was found that the system offered huge performance advantages for model execution and calibration when compared with a non-distributed version of the software. However, it was found that display operations ran significantly slower than on a standalone machine due to inherent limitations of the high level interapplication communication technology used (Apple Events). Methods of improving the performance of such operations were suggested.

Finally observations regarding the benefits of the design of GeoAnalyser were made. It was concluded that the demonstration system generally offers very useful features for the tasks which it was designed for. Some areas of improvement were also suggested.

Chapter 6

Implementation Issues of Intelligent Spatial Decision Support Systems

This chapter summarises the user interface and implementation issues arising from the development and application of the GeoAnalyser system. Issues concerning the integration of Intelligent Systems techniques within ISDSS are examined. The main design and implementation issues surrounding the development of ISDSS are presented. This provides the basis for developing a foundation for ISDSS.

6.1 Introduction

The previous two chapters have described the detailed design of the GeoAnalyser ISDSS and its application to the problem of industrial modelling. This chapter takes stock of both the explicit and implicit design and implementation issues raised during the course of developing the system and applying it to a particular problem domain. An assessment is made of the integration issues involved in using Intelligent Systems techniques for ISDSS as well as general implementation issues surrounding the development of ISDSS.

6.2 Findings From Development of GeoAnalyser System

From the development and application of the GeoAnalyser system, the following observations can be made:

- *Flexible user interface*. The user interface provides a great deal of flexibility in that multiple windows containing any of the three supported display tools depicting any data within the data set can be displayed and arranged as required. This allows the interface to be adapted to the user's preferences and aids in the overall effectiveness and usability of the system.

- *Client-server techniques*. In GeoAnalyser, client-server methods successfully provide a means of reducing the computational overhead of a complex spatial model on the local machine. However, it is clear that the effects of peak network speed (there is clearly a threshold below which it becomes infeasible to use these methods) and latency and other constraints either in the operating system or the network can be considerable (for example, the hard upper limit on the rate of communication of Apple Events). It was found that these effects are particularly

noticeable when they have a direct impact on user interface operations. It was concluded that caching can help considerably in those operations where the spatial data remains unchanged.

- *Exploratory analysis*. The user is free to choose any system operation at any stage. This significantly helps the user to explore the problem domain. This flexibility also has the benefit of widening the range of application of the system.

- *Plug-in model architecture*. This extends the flexibility of the system by providing a limited form of model base management. The architecture facilitates the parallel development of spatial models, with one of several alternative models being selected before the system is run. Such an architecture also lends itself to the development of models by third parties.

- *Comparison tools*. Presently only one model or path of evolution may be performed using the system. This stems from the linear spatiotemporal data structure employed. It was found that this inhibited direct side-by-side comparison of models (where the results of running two or more models are compared) and model evolutions (where the same model is run based in a different set of initial conditions). This could be accomplished within the system using either an alternative data structure (such as an n-ary tree) or support for multiple data structures (analogous to the way that several documents may be open at once in a word-processor). Tools for these complex structures would also be required.

A further observation of the system is that the spatial model development and calibration phase are a very different type of activity from the spatial model deployment phase. While GeoAnalyser provides facilities for both aspects of spatial modelling this might not necessarily be appropriate for either model developers or end users. A related issue concerns the integration of intelligent techniques within the ISDSS. This is discussed in the next section.

6.3 Implementation Issues of Intelligent Systems Techniques

Intelligent systems techniques tend to operate in two distinct phases. During the *development phase*, a greater degree of interaction is required for building and testing the system. For example, with feed forward neural networks used in supervised learning, network topology (number of input neurons, output neurons, hidden layers) and learning parameters are determined and the network trained using the learning dataset during this phase. Once a satisfactory internal model has been built it can be deployed in the *application phase*. Taking the example, the trained neural network would be presented

with unseen data from the target problem domain. The different levels of interaction required during each of these phases are summarised in Table 6.1.

Technique	Development Phase	Application Phase
Neural networks	Access to topology, learning rate, threshold, activation function, transfer function. Requires representative test data set if supervised learning is used.	Recall phase – requires access to test data values which the trained network uses to generate the result.
Genetic Algorithms	Settings include number of pools, individuals, gene structure and representation, genetic operators (including access to their rate parameters), fitness function and also data required to evaluate the fitness of an individual.	Data required to evaluate the fitness of an individual.
Fuzzy Logic	Form of fuzzy membership functions, rules stored in the rule base, inference mechanism.	Interactive access to knowledge base through query interface.
Expert Systems	Rules stored in the rule base, inference mechanism.	Interactive access to knowledge base through query interface.

Table 6.1 – Interaction Requirements of Intelligent Systems Techniques

In GeoAnalyser, detailed development of the genetic algorithm is hidden from the user. Therefore, the fitness function and gene structure are hard-coded into the system. However, the user must still specify the genetic operators to be used, their associated rate parameters and also the configuration of the population.

It can be argued that these issues are similar to those associated with the integration of traditional spatial models and tools within SDSS. For example, regression can be considered to involve a development phase during which the regression coefficients are calculated. The difference between these methods and intelligent techniques is that, in the case of the latter, it is very difficult to completely automate the development phase whereas with the former, default parameters can usually be assigned if there is no obvious method of automation. To a certain extent, it also depends on the expectation of the users i.e. does the user expect to have to specify the detailed set up of a neural network or can at least part of this be handled automatically (using for example a rule base endowed with knowledge of the problem domain).

Aside from these issues, other implementation issues are very similar across all types of analysis and modelling techniques (e.g. user interface, system design, other aspects of integration etc.).

6.4 Applications Versus Toolkits

In any particular spatial problem domain, there are two extreme types of software tool that can be built. *Problem-specific tools* provide support for addressing particular kinds of problem e.g. transport modelling. In this case generic techniques (covering traditional modelling and statistical tools as well as intelligent techniques) will be used to solve particular parts of the overall problem. In this respect, such tools can really be described as *applications* of generic techniques.

At the other extreme are *toolkits* or *application generators* which are not geared to solving any particular problems but provide broad support for a whole range of different problems. In such environments, generic tools are available to help the model builder develop an understanding of the problem domain and assemble the necessary application by adapting existing tools and techniques.

In the case of GeoAnalyser, the role of the genetic algorithm is absolutely clear in that it can only be used for evolving model parameters. However, by developing alternative types of model (using the plug-in architecture), different types of problem can be addressed. So GeoAnalyser has certain features of both an application and application generator.

Generally speaking, there is a trade-off between features and complexity. The exact mix of these will depend highly on a combination of the user's expectations and the nature of the problem domain.

6.5 Implementation Issues of ISDSS

The above discussion has revealed that the nature of intelligent techniques is distinct from other modelling and analytical methods commonly used for spatial analysis. What then are the overall implementation issues involved in building ISDSS? Generalising from the above observations, these can be summarised as follows:

- *Adaptability to problem domain.*
- *High level problem solving support.*
- *Flexible and friendly user interface.*
- *Support for exploratory analysis.*
- *Effective use of distributed resources.*
- *Speed of operation.*

- *Design and implementation autonomy.*

- *Reuse of components*. For example, the use of a standard GIS for the display subsystem (discussed further in the next chapter). This can have many benefits, including simplification of system design and reduced development costs.

- *Access to control methods of intelligent system techniques*. For example, for setting up the initial population, operators and parameters in a genetic algorithm as well as providing interactive control of the evolution.

The next stage is to examine some of the possible architectures that could be adopted for ISDSS bearing in mind the issues described above. This is undertaken in the next chapter.

6.6 Summary

This chapter has discussed the issues uncovered during the development and application of GeoAnalyser as well as integration issues surrounding the deployment of intelligent techniques within ISDSS. Differences between the development of applications and toolkits were also discussed. From these considerations as general set of implementation issues involved in building ISDSS was presented.

Chapter 7

Implementations of Intelligent Spatial Decision Support Systems

This chapter presents general architectures for building ISDSS using distributed computing techniques. In each case, the system construction is described in detail. The architectures are assessed in terms of the level of user interface integration, data integration and problem solving capability.

7.1 Introduction

The basic client-server model of distributed computing has been discussed in Section 2.5.1. In this model communications involve an exchange between a single client application and a single server application. This simple picture, however, belies the complex structure that distributed systems can exhibit. For example, a client requesting a service from another application could be simultaneously acting as a server for a third application.

This chapter explores various structures that are suitable for building ISDSS (first published in Sandhu and Treleaven 1996). As discussed in the previous chapter, while there are implementation issues that are peculiar to ISDSS, other more general implementation issues are common to both ISDSS and SDSS (for example, use of distributed computing, system extensibility, decision support functions etc.) Therefore the architectures proposed here are also applicable to a much wider class of system.

The architectures presented here are as follows:

- Simple Client-Server Model
- Three Tier Model
- Workbench Model
- Data Dispatch Model

Each of these architectures is comprised of the four functional modules discussed in Section 2.2 (i.e. the database management system, analysis routines, display and report generators and the user interface), although these modules are not necessarily reflected in the actual system organisation.

Each architecture is described in detail together with variations on the basic structure proposed. The criteria used for evaluating the architectures are as follows:

- Ease of information exchange
- Flexibility and effectiveness of user interface
- Speed of operation
- Design autonomy
- Use of distributed resources
- Problem-solving capabilities

Before discussing the architectures, there are some important issues common to most of the architectures concerning the display subsystem that are addressed in the next section.

7.2 Mapping and Display Subsystem

Whereas the analysis/modelling subsystem and other components of an ISDSS are usually custom built for a particular problem domain, there is a clear choice for the mapping component. Either an off-the-shelf GIS could be used or it could be custom-built. Building a custom display system is usually the least favoured route for two reasons. First, the functionality required for most applications already exists in pre-built software (resulting in much lower development costs). Secondly, GIS generally provide extensive mapping tools surpassing the requirements of most ISDSS (the exception being those cases where time plays an essential part of the analysis). These tools would be hard to match with a custom component.

Whilst GIS have most of the features required of a spatial data display system, this is not the only factor affecting their use as components of ISDSS. Of paramount concern for the distributed architectures that are presented in the sections to follow is the ability of the GIS to co-operate within the larger framework of the ISDSS using interapplication communication. However, as noted in Chapter 2 there is a large disparity in the nature of the interapplication communication interface provided by existing GIS. The various types of interface offered by GIS are compared in Table 7.1. Custom built mapping systems are also included in the Table for comparison. The factors used for the comparison are as follows:

- *Ease and speed of data access.* How easily and rapidly an external component of the ISDSS can extract and send data. This is determined by several factors, including the process of converting between different data structures (including running any macro routines inside the GIS), speed of the underlying GIS database, and the IAC throughput of the operating system (which is affected by the speed of

the computers concerned and the speed of the network linking them if the components reside on different physical machines).

Class of System	**Ease and speed of data access**	**Ease of access to control mechanism**	**Integration with other subsystems**	**Focal point of user interface**
Custom built viewing system (i.e. not GIS based). Includes spatial visualisation code libraries.	Very easy and fast	Potentially full access	Excellent, resulting in seamless system	Any subsystem
GIS with macro language featuring user interface extensibility and IAC interface	Very easy and fast	Depends on IAC interface	Good, but seams between the systems may be visible	Any subsystem, depending on nature of IAC interface
GIS with full IAC interface for displaying, retrieving and storing spatial data	Very easy and fast	None	Good, but seams between the systems will be clearly visible	Must be another subsystem
GIS with IAC interface for data access only	Very easy and fast	None	Poor, with seams visible	Split between GIS (for displaying spatial data and results of analysis) and the other subsystems.

Table 7.1 – Comparison of Classes of View Subsystem for ISDSS

- *Ease of access to control mechanism.* The ability to integrate the subsystems of an ISDSS not only at the data level (so that data can pass freely between the various subsystems) but also the *control level* is critical for integrating the systems seamlessly. In order to achieve tight user interface integration, the various subsystems need to be able to access procedure calls within other related subsystems and this requires some form of hook into the control mechanism (usually in the form of an event processing loop). Clearly, if IAC is to be used as the glue between subsystems then at least part of the control mechanism should be accessible through the IAC interface.

- *Integration with other subsystems.* This assesses how well the combination of the above two criteria affects the level of integration between the subsystems of the ISDSS.

- *Focal point of user interface.* Usually in an integrated system, one of the subsystems tends to assume greater control of the user interface, with the other subsystems being delegated particular tasks that feed into the main flow of control in the system. It is important to determine the various options available to the system designer for creating the user interface according to the class of GIS concerned.

Of course a precondition for employing GIS as the mapping subsystem is that they must actually provide the display functions required in the ISDSS.

As shown in the Table, the greatest level of integration at both the data and control level is achieved by custom display system, as these aspects can be built exactly to requirements. However, the flexibility offered by GIS endowed with a macro programming language and extensive IAC interface can achieve a very similar result. Depending on exactly how extensive the IAC interface is (and in particular whether the IAC features are accessible directly from the macro language), such GIS can theoretically form the overall framework of an ISDSS if required or they can simply provide the display subsystem. MapInfo is a good example of such a rich system which provides a programming language derived from BASIC, called Map Basic, for user programs. However, the flexibility that is provided by macro programming languages is achieved at the expense of speed as these are usually interpreted rather than compiled into machine code.

For GIS that do not have these features, the only recourse is to use them for the display subsystem. Without the necessary hooks into their control mechanisms, these GIS cannot be well integrated at the user interface level with the rest of the components of the ISDSS.

These observations will be used to discuss and evaluate the ISDSS architectures that are described in the following sections.

7.3 Simple Client-Server Model

Using the client-server model introduced in Chapter 2, one can construct the system shown in Figure 7.1. Here the modelling and display systems are linked using direct co-operative coupling (see Section 3.5.3 for a definition). A GIS could be used for the display system, as suggested by the Figure. Data and control messages are exchanged between client and server using the IAC mechanism.

The nature of the integration that can be achieved in systems constructed based on this architecture depends very much on the class of the GIS used, as described in the previous section. Using a more sophisticated GIS with a macro programming language and

extensive IAC interface, one could construct a system that is very tightly integrated at both the data level and the control level, thereby creating the illusion of a single unified user experience. Alternatively, the system could be one that is integrated at the data level only, with the less well featured GIS acting as a map display subsystem and the modelling server providing most of the control functionality for the overall system.

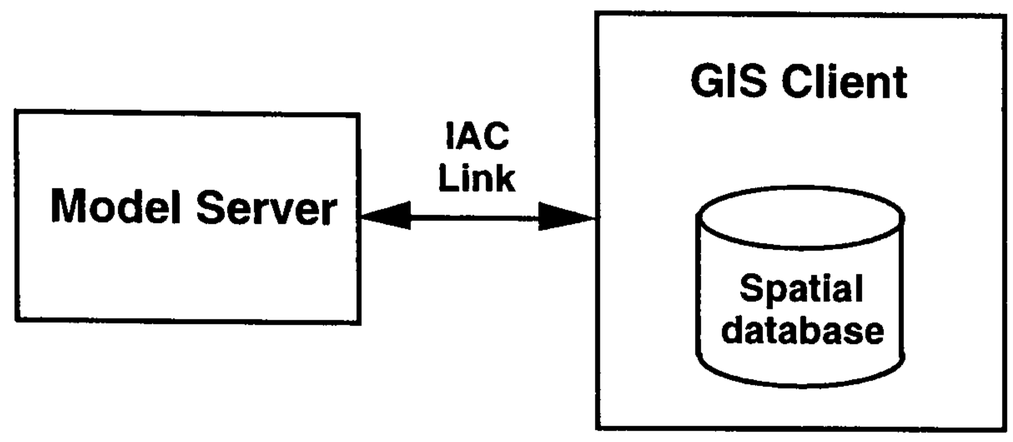


Figure 7.1 – Simple Client-Server Model

This flexibility arises mainly from the simplicity of the arrangement, as the model server can be expanded to become the focal point of the user interface (as discussed in Section 7.2) according to the system requirements if the class of the GIS client warrants it. This aspect is covered in much more depth in the Workbench Model.

The system might typically be used by selecting the appropriate data from within the GIS and choosing the type of model to run before performing the analysis. This task would be carried out through the macro language. At the request of the user, the appropriate macro subroutines are executed, which extract the required data from the spatial database and send them to the model server, where the results are generated and returned to the subroutine for display.

Applying the assessment criteria outlined earlier:

- *Ease of information exchange*
  Even for the most basic class of GIS discussed earlier, information can be exchanged fairly easily using IAC messages (this requires the spatial data structures to be packed as a byte stream into the IAC message). Additionally, some form of data conversion at the server end will probably be required. A drawback of this architecture is that if a different display system were to be used then the server communications and data conversions would have to be reworked; functionality that is hard-coded into the server. Furthermore, the architecture relies on all permanent data to be stored either at the server end or at the client end (in the internal database of the GIS). This means that initially, at system start-up time, data needs to be exchanged between the client and the server. During the course of using the system, both the client and server temporary data stores need to be kept synchronised so that, for example, maps drawn in the GIS accurately reflect

attributes that have been calculated using the analytical tools provided by the server. This creates an additional overhead that could be quite limiting where large volumes of data are concerned.

- *Flexibility and effectiveness of user interface*
  Even with this simple architecture, a powerful and effective user interface can be developed, with either the client or server taking on the role of the decision support framework depending on the sophistication of the GIS. Some sophisticated operations, such as linked point and click operations between the client and the server (for example to perform a calculation on a particular point in a region), require exposure to the control mechanism of the GIS. Without this level of access even simple interaction between the client and server is not possible at the level of the user interface.

- *Speed of operation*
  As there are only two subsystems involved, the system can be made quite responsive. The speed is limited mainly by the speed of the IAC links, the need for synchronised data sets and also the execution speed of the GIS macro language (if used).

- *Design autonomy*
  As mentioned above, a number of features need to be hard-coded into the system, considerably reducing the autonomy of the individual subsystems.

- *Use of distributed resources*
  The separation of the analytical components of the ISDSS from the display system can considerably improve the responsiveness of the system. However, the overhead of keeping distributed subsystems in step can reduce these benefits.

- *Problem-solving capabilities*
  The capabilities of the system for solving complex spatial problems is limited only by the functionality of the model server and the level of integration that can be achieved. With sophisticated GIS, quite intricate systems can be built. With more basic classes of GIS, the model server can take on a more prominent role, even if the user interfaces cannot be linked at the operational level (such as for linked point and click operations).

It will be noted that the GeoAnalyser system described in Chapters 4 and 5 is essentially based on this simple architecture. In this system, the need for building specific tools for visualising the spatio-temporal dynamics of a dataset required that a custom display system needed to be developed. While this simplified many of the communication issues

involved in integrating the GeoClient with the GeoServer, two custom systems needed to be developed rather than one. This also raised an opportunity to minimise synchronisation problems by using specific IAC messages to control the state of the two systems.

Furthermore, it was decided that GeoClient would form the focal point of the user interface, leaving the processing intensive modelling operations to be dealt with by GeoServer. This strategy resulted in a very responsive user interface while retaining the advantages of using distributed computing. Except for the initial step of linking the GeoClient to a GeoServer, the distributed architecture is completely transparent to the user.

7.4 Three Tier Model

Three tier systems are increasingly used in corporate computing where the middle layer abstracts the business rules or business logic that would otherwise be embedded in the database management system (as stored procedures) and client application. These business rules, which often change within the lifetime of the system, can be modified without requiring other parts of the system to be redesigned while in most cases improving the performance of the system by making better use of distributed processing. This abstraction permits other components to be considerably simplified and they can be modified independently of the other components. For example, changes to the business rules will not usually affect the structure of the database

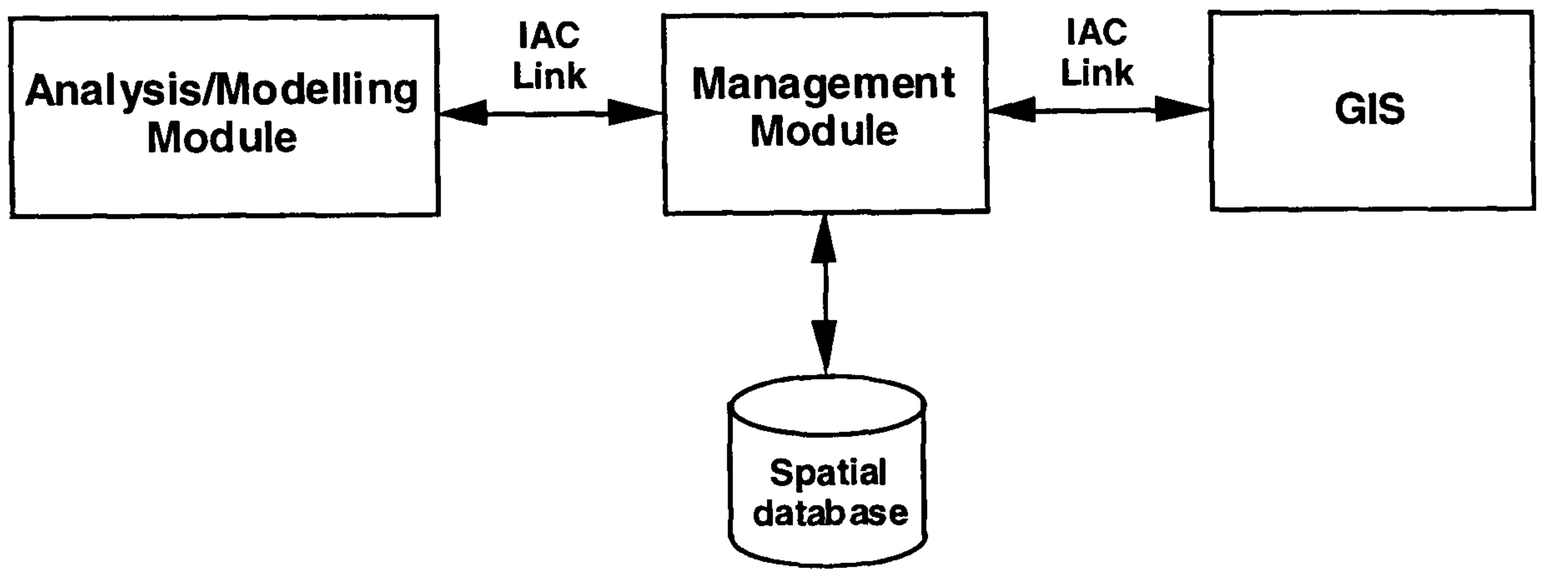


Figure 7.2 – Three Tier Model

The Three Tier Model, depicted in Figure 7.2, builds on the design of the Simple Client-Server Model. The main difference is the use of *indirect co-operation* to include a third interacting subsystem, the management module, to provide an abstraction layer similar to that achieved in corporate three tier systems. This layer provides scope for the following facilities:

- Decision support framework for high level operations (for example to enable comparison between model runs, saving model results etc.). The framework could

also provide support for guiding users through more complex operations or set procedures that must be applied in a particular sequence. For example, assistance could be provided for changing the configuration of a neural network in the analysis/modelling module. This could also include help facilities for inexperienced users by guiding them through typical operations.

- Managing configuration details of the analytical module (for example model parameters, neural network topology etc.).

- Providing simultaneous access to the spatial database. This eliminates the problem associated with the Simple Client-Server Model (described in Section 7.3) of maintaining separate synchronised datasets in the display and modelling subsystems). A single database is used for both modelling and display purposes. Access to the database can be arbitrated through the management module.

- Providing expanded options for customising the communication between the analysis/modelling module and the GIS. This mirrors the use of the middle layer to house and manage the business logic of corporate three tier systems. In an ISDSS this layer might be used to perform the data conversions between the other two modules, for example.

- Potential for providing support for multiple models. This can be achieved by linking with multiple modelling modules and using some form of internal model base management.

The system is assessed in terms of the criteria outlined earlier:

- *Ease of information exchange*
  As with the Simple Client-Server Model, information can be exchanged quite easily using IAC messages for even the most basic class of GIS. Data conversions can be performed by the management module. While these conversions are still hard-coded into the management module, a change of display system could be incorporated without the need for changes to the analytical subsystem. All permanent data is stored in the single database, even though local copies may also be kept by, for example, the GIS. The advantage of this is that, in effect, a single global copy of valid and complete data is maintained. Normally display operations only require access to a subset of the data and so speed should not be a problem here. However, the analytical module often requires access to a much larger subset of the data and so heavy use will be made of IAC messaging to retrieve and update this data prior to and following analytical operations. This could be alleviated to some extent by using local data caching schemes.

- *Flexibility and effectiveness of user interface*

  The management module can provide a central point of control for the user interface. In this sense, the analysis/modelling subsystem and the GIS can effectively act as servers to the management module. As with the Simple Client-Server Model, the sophistication of the user interface operations between the subsystems depends on the nature of the IAC interface provided. Specifically, linked operations between the GIS and other two subsystems require access to the control mechanisms of the GIS at the IAC interface.

- *Speed of operation*

  With three linked subsystems, response times can be a problem, particularly when transferring large amounts of data to or from the database. However, provided the IAC links are reasonably fast, the greater potential for using distributed processing should result in a sufficiently responsive system. If data transfer becomes a problem, local data caching schemes can be employed.

- *Design autonomy*

  Use of the management module means that changes to the design or operation of the system should not require changes to all parts of the system.

- *Use of distributed resources*

  Provided the bulk of the user interface is contained within the one system, then use can be made of distributed processing resources (with each module potentially residing on independent processors). However, if the user interface is built from both the GIS and the management module , then these modules really need to reside on the same machine.

- *Problem-solving capabilities*

  The architecture provides an effective architecture for solving spatial problems. Even if the GIS has only a rudimentary IAC interface, the more complex operations can be handled directly by the management module. The management module can also provide a convenient framework for high level decision support functions, such as for comparing results and for model base management. With more sophisticated GIS, linked operations with the other subsystems can be achieved, which greatly helps to create a flexible and powerful user interface that can be used to solve ill- and semi-structured spatial problems.

7.5 Workbench Model

The Three Tier Model improves on some of the problems associated with the Simple Client-Server Model by the introduction of another interacting subsystem. The

Workbench Model uses direct co-operation (as with the Simple Client-Server Model) but expands considerably the functionality of the analysis/modelling component of the Simple Client-Server Model. The diagram in Figure 7.3 outlines this architecture.

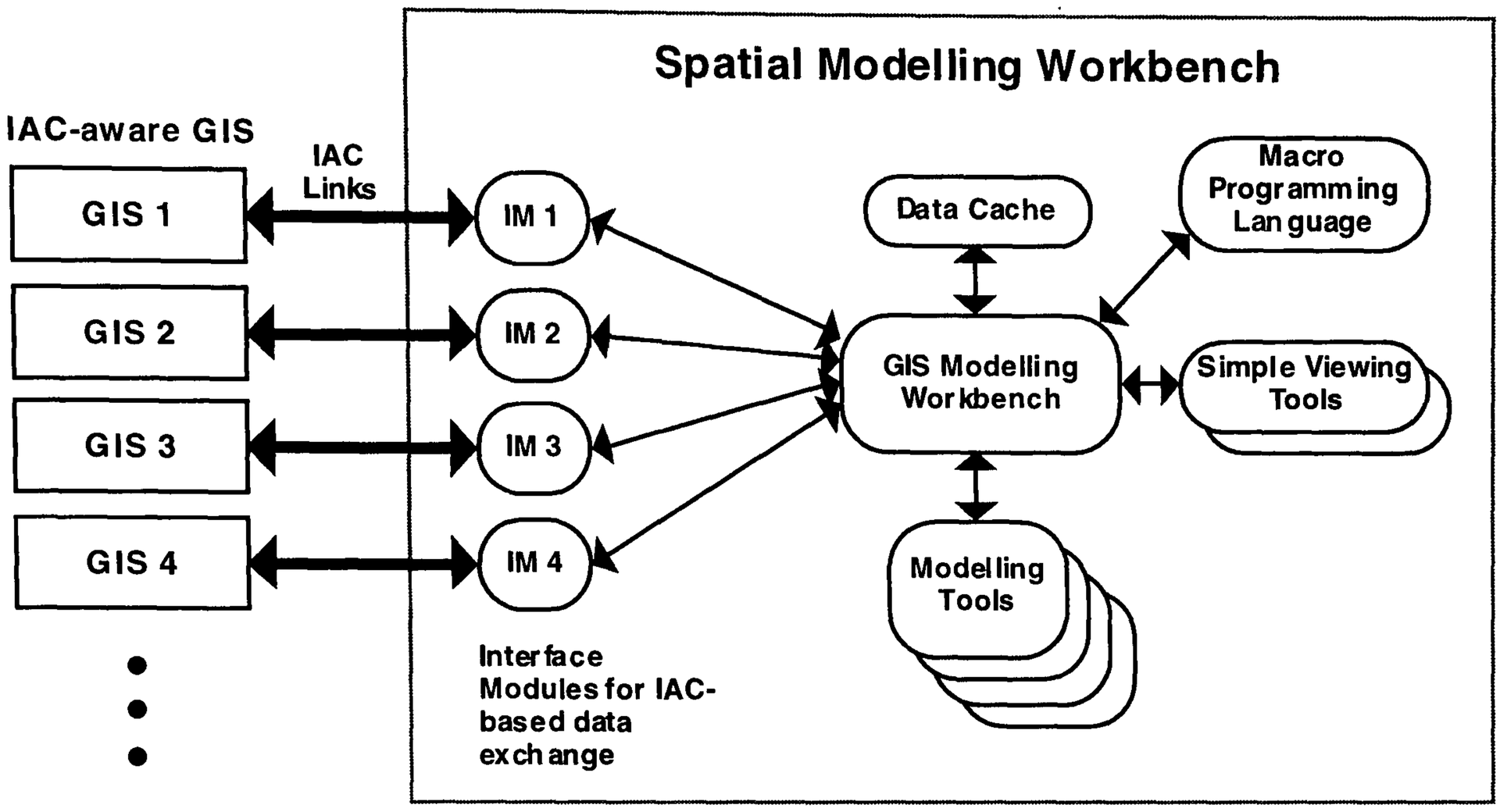


Figure 7.3 – Modelling Workbench

The workbench would be implemented as a single object-oriented application that can co-operate with IAC-capable GIS (which can obviously exist anywhere in the local network domain). In this scheme the (usually good) spatial data management functionality of the GIS would be used to supply the Workbench with data. Once initiated, all of the user interaction would take place within the Workbench, which would provide specific tools for spatial analysis. The Workbench has the following subcomponents:

- *Interface modules*
  In a similar way to hardware device drivers, these abstract the details of a particular GIS (or other display system) and also perform the task of converting spatial data from one format to another and enable the Workbench to be integrated with a number of GIS (though not simultaneously). An Application Programmer's Interface could be developed for creating additional modules for other GIS. Such conversion tools have been developed as standalone systems (Pascoe and Penny 1990) and it would be fairly straightforward to integrate this within the Workbench.

- *Workbench module*
  This is the key subcomponent and is responsible for co-ordinating the activities of other objects in order to create a unified user interface. The Workbench module would execute scripts (written using the built-in macro language) and organise the sending and receiving of data between all of the

subcomponents. It would also maintain an internal data cache to minimise the need for data requests to the GIS, thereby improving the efficiency of the system. All of these activities would be performed in the *event loop*, which is the main control mechanism of the Workbench.

- *Modelling tools*
  This library of objects would perform the analysis on data fetched by the Workbench module from the GIS. The library of modelling objects might include specialised objects such as genetic algorithm-based search and optimisation modules, pattern recognition modules using neural networks, location-allocation models for shopping, transport models etc. as well as basic building blocks for ISDSS (neural network, genetic algorithm, rule based systems). Links could also be implemented between these model objects and the viewing tools to display results immediately and also allow for complex user interaction (such as point and click operations that feed point data back into the model). An API could be developed to permit developers to add their own model objects.

- *Viewing tools*
  Standard view objects such as thematic maps, graphs etc. would be provided. Custom views could be added through the provision of an API. Linked views could also be supported by embedding hooks into the control mechanism of the Workbench.

- *Macro programming language*
  Although not strictly necessary, a simple scripting language is very useful for customising and automating spatial operations and model procedures. By integrating the programming language with the other elements of the Workbench, every aspect of the system could be made scriptable.

- *Data cache*
  The data cache would reuse data fetched from the GIS in earlier operations and is vital to improve the overall efficiency of the system.

Applying the same assessment criteria as before:

- *Ease of information exchange*
  Only basic data reading and writing operations are required between the GIS and the Workbench. This should be possible even for the simplest GIS sporting IAC interfaces. The problem of data conversion is dealt with through the Interface

modules. Therefore, information exchange should not present a problem as far as the logistics are concerned.

- *Flexibility and effectiveness of user interface*
  A very powerful and flexible user interface can be constructed for the Workbench since all user interface operations are conducted through the one system. This enables sophisticated operations that involve messaging between different objects within the Workbench (such as point-and-click operations or the dynamic display of model results generated by model objects within viewing objects). Furthermore, the macro programming language should enable users to modify key aspects of the system according to need.

- *Speed of operation*
  As with the Simple Client-Server Model, there are only two subsystems involved. The operating speed of user interface operations should therefore be very high as no IAC messaging is involved directly. However, operations that require data to be retrieved from or written to the GIS will be slowed down by a factor which depends on the complexity of the data conversion in the Interface module, the speed of the IAC link and also the time taken by the GIS to process the data request. The extent of this process is minimised by the use of the data cache. Therefore, the speed of such operations should increase as more data becomes cached.

- *Design autonomy*
  The Workbench itself is quite independent of the GIS. However, the objects that make up the Workbench are very much dependent on the design choices made in the Workbench. If object oriented design is used, as suggested here, then these objects can be developed fairly independently of one another so that internal changes to the design of a particular object should not have ramifications elsewhere in the system.

- *Use of distributed resources*
  This architecture does not make very effective use of distributed resources, for example for the model objects. This means that the system would require a fast machine to run on.

- *Problem-solving capabilities*
  The system should provide excellent problem-solving capabilities that are not limited by the type of GIS that is linked with the Workbench. The main features are a flexible, powerful and highly interactive user interface, an extensible library of modelling tools, macro programming language for creating custom applications

and modifying the default behaviour, and built-in viewing tools for visualising data and model results. As most aspects of the system are extensible through APIs, the Workbench should be a very versatile form of ISDSS. However, all this is achieved by weakening the link with the GIS, which is used mainly for it's strengths in storing and handling spatial data.

7.6 Data Dispatch Model

The Data Dispatch Model (illustrated in Figure 7.4) adopts an alternative strategy to building a sophisticated ISDSS while still making use of distributed computing methods by considering the design of the GIS as an integral part of the problem, rather than providing links with external systems. The system uses a hybrid form of indirect co-operation internally within the system to implement distributed objects. However, the system is self-contained and operates independently of any other application (although creating such links would be fairly straightforward; for example to import live data from a GIS as with the Workbench Model).

This architecture uses object oriented design principles to develop complex ISDSS that provide a high degree of design autonomy for the system components. The architecture is based on the concept of *participants*. These are objects that have an active role to play in the system event mechanism. Each participant would be derived from the participant base class that provides basic behaviour common to all the participants. This includes the ability to respond to and generate system events. These events can trigger actions in other objects which in turn can generate other events and so on. This allows for quite complex linked behaviour to be created.

The system is comprised of the following components:

- *Database Management System*
  Some form of database management system is usually the best method of managing spatial data, performing queries on that data and maintaining strict access control and data integrity throughout all operations. As the system itself is object oriented, it would be more efficient to use an Object Database Management System (ODBMS) rather than a Relational DBMS (RDBMS).

- *Data Participant*
  This interfaces with the data source and the rest of the system. Being a participant object enables it to respond to and generate application events (see below). It would also contain a cache to help improve data access efficiency.

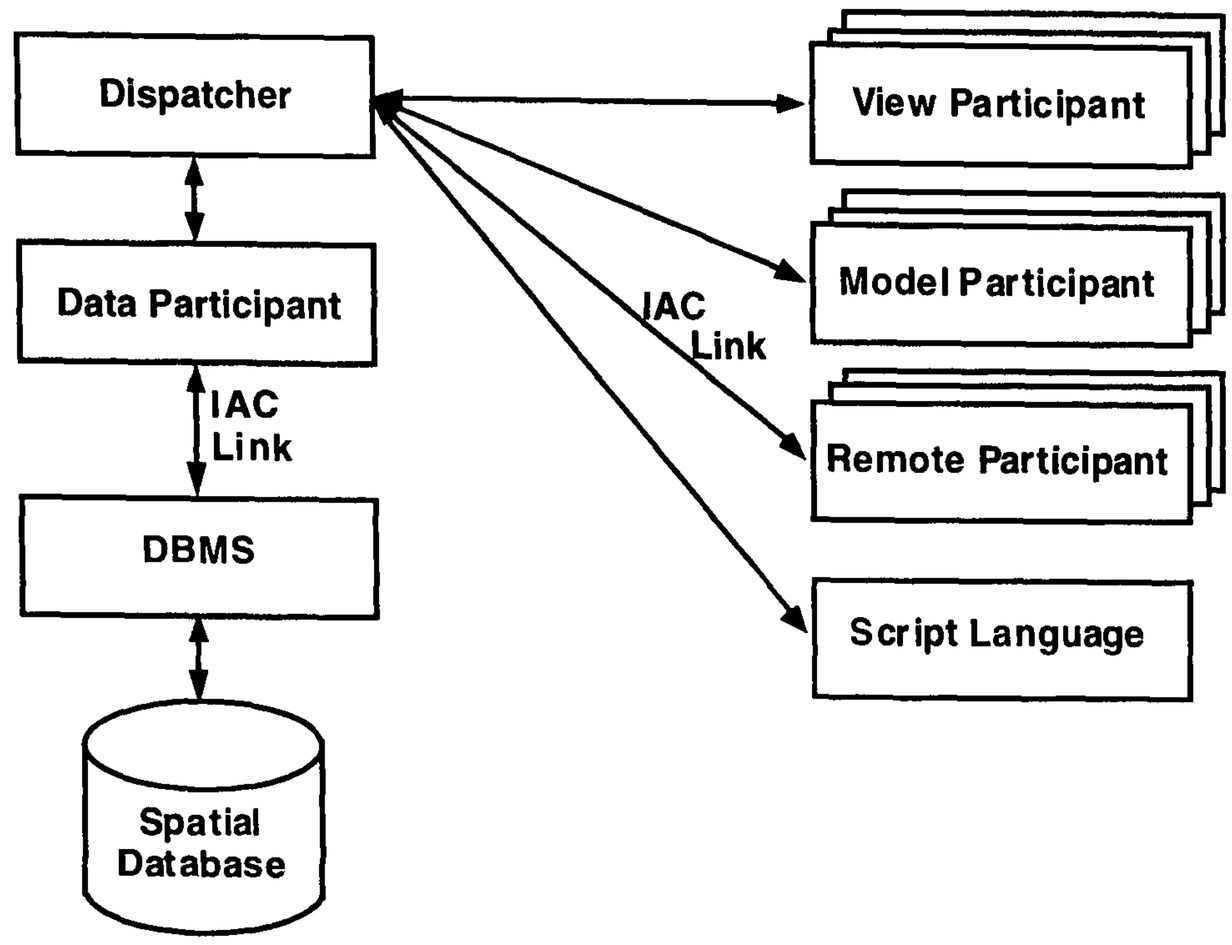


Figure 7.4 – Data Dispatch Model

- *Dispatcher*

  This object is responsible for the main control mechanism of the system. It maintains a list of active participants (which may be spread over a network) together with details of what object type they are and their position in the activity list. The dispatcher would ensure that events get delivered correctly to the right participants and in the right sequence. It also permits participants to send messages to each other. For example, if a certain data item is to be altered, then the participant making the change would notify all others to update their state accordingly. An elaboration on this scheme could restrict messages to certain types of participant (for example, to limit a message to all objects of type 'view'). This object should not be confused with the dispatch object that is usually responsible for managing the event loop behind the graphical user interface of an application as although the two objects are similar, the type of event processing is quite different.

- *View Participant*

  This would implement an object that displays data in a graphical form. Views supported might include thematic maps, graphs, tables, animations, 3D views etc. Each view would handle standard messages (such as for updating the currently displayed image) and more specialised ones (e.g. to select a certain set of data and show the active selection). Linked views would be supported by exchanging select and update messages between associated views.

- *Model Participant*

  This would implement the required analysis functions. A library of models could be developed as with the Workbench Model, including basic objects for intelligent

computing methods. In addition to performing user interaction and issuing update messages to views, model participants also have the ability to create new data sets. Together this means that a model participant can be fully integrated with a view object and respond to events generated by the view object (such as when the user clicks in a region on a map).

- *Remote Participant*
  This is a participant that is derived from an alternative participant base class that differs in that all communication is performed using IAC methods rather than by direct procedure calls. This allows for the construction of data or model participants that are fully integrated with the system at both the data and control level but are actually located on remote computers. For example, a processing intensive model participant could reside on a separate machine from the rest of the system yet still play a full role. This would provide an opportunity to build a fully distributed system that has none of the communication problems that are involved in linking with existing systems using IAC. Providing a base class for remote participants ensures that the construction of these objects is as straightforward as other objects in the system.

- *Macro Programming Language*
  As before, a script language would facilitate the construction of custom systems and automate routine tasks, thereby endowing the system with a high degree of flexibility. Each of the objects in the system (including remote participants) would be accessible through the programming language.

Applying the assessment criteria outlined earlier:

- *Ease of information exchange*
  As the system is, with the exception of the DBMS, entirely custom built, there is therefore no need to convert data from one representation to another. Furthermore, the remote objects are tightly integrated with the other objects in the system; local objects should present no difficulty. Therefore there are theoretically no problems associated with exchanging data between the objects of the system.

- *Flexibility and effectiveness of user interface*
  As the system is entirely custom built, a highly developed user interface can be provided. The system event model that is proposed (which incorporates different levels of involvement in the control mechanism of the system) would allow for complex linked operations between participants which would greatly aid in the exploration of spatial problems.

- *Speed of operation*

  The use of remote participants gives rise to two main speed problems. First, as the data participant is conceived as a remote object, data read and write operations would be limited by the speed of the IAC links. However, the model and view participants could maintain a data cache to help speed up data access (data that has already been retrieved would be read directly from the cache rather than using slower IAC messages to read them from the data participant). An alternative strategy to combat this problem would be to make the data participant a local object, thereby removing the reliance on IAC messaging for this critical component. Secondly, remote model participants that require access to large volumes of data would be considerably slower than local ones. While caching would help to some extent, this is not a solution when the data is continuously changing. The only real solution to this is to construct such models as local objects.

- *Design autonomy*

  Use of object oriented methods should result in system objects that are quite independent of one another as far as their inner workings are concerned..

- *Use of distributed resources*

  As with the Workbench Model, the ability to locate the analytical/modelling components of the ISDSS on faster computers on the network can improve the speed of the system. Clearly an overall improvement would only be possible if the overhead associated with IAC messaging was outweighed by the gains made through using distributed resources.

- *Problem-solving capabilities*

  The system should provide an almost ideal framework for the development of ISDSS that are fully integrated and provide highly interactive user interfaces sporting a selection of viewing tools and analysis/modelling tools.

This system incorporates all of the components required for building an ISDSS in its own right. The main criticism of adopting such a radical design is that few existing components can be used, giving rise to high initial development costs. However, once such a system has been built and a suitably rich API developed (offering full integration for external analysis modules), highly effective and extensible end user applications can be developed.

7.7 Comparison

Each of the architectures described takes a different approach to building ISDSS. These approaches are summarised in Table 7.2. While the architectures and combinations of design features are by no means exhaustive, they do illustrate the implications that the following important design issues have for ISDSS construction:

- integration of GIS as part of the ISDSS variously as the main application framework, display system and spatial database;
- distributed computing methods that rely on interapplication communication techniques as a means of integrating the distributed components of the system;
- multi-tiered client-server architecture;
- custom-built components;
- complete custom-built systems that may interact to some extent with external applications.

Architecture	Method of Integration	Description of Approach
Simple Client-Server Model	Direct Co-operation	Composed of model server and GIS client. Uses IAC methods to integrate subsystems.
Three Tier Model	Indirect co-operation	Composed of modelling module, management module (for decision support framework, configuration management, controlling database access, managing communication between modelling module and GIS, model base management), GIS and database. Use IAC methods to integrate the subsystems. Database is directly linked to management module.
Workbench Model	Direct co-operation	Enhanced form of Client-Server Model. Composed of GIS linked with the Workbench. The Workbench provides a rich environment for decision support, modelling and viewing spatial data.
Data Dispatch Model	Hybrid direct/indirect co-operation	Complete custom system that requires no external systems. Provides rich environment for building ISDSS with a sophisticated user interface composed of linked models and objects. Supports development of fully distributed ISDSS.

Table 7.2 – Comparison of Design Approaches Used for ISDSS Architectures

The evaluation of the systems over the previous sections that was performed according to the criteria that were defined in Section 7.1 is summarised in Table 7.3.

Architecture	Information exchange	User Interface	Operating Speed	Design Autonomy	Distributed Processing	Problem-solving capabilities
Simple Client-Server Model	•	●*	●	•	●	●*
Three Tier Model	●	●*	●	●	●	●
Workbench Model	●	●	•	●	•	●
Data Dispatch Model	●	●	●	●	●	●

* Depends on the sophistication of the GIS' IAC interface

Table 7.3 – Assessment Summary of ISDSS Architectures

The Simple Client-Server Model is capable of creating reasonable ISDSS. However such ISDSS are deficient in two key areas: the ease with which data could be exchanged between subsystems and the design autonomy of the client and server. Two of the remaining criteria (both related to the user interface which can be developed) were dependent on the existence of a fairly sophisticated IAC interface between the GIS and the model server.

The Three Tier Model alleviated some of these difficulties through the introduction of a third middle layer. This yielded a greater level of control and design autonomy than was possible using the Simple Client-Server Model. However, the integration of the user interface between the three tiers in the system was still dependent on the sophistication of the IAC interface provided by the GIS. The conclusion is that the ability to integrate the user interface between distributed subsystems is dependent on the existence of a suitably complex IAC interface between communicating components. Where the overall system is composed partly from custom built components and partly from off-the-shelf software, the IAC interface of the latter is a critical factor in determining the level of integration that can be achieved for the user interface.

The Workbench Model attempted to resolve some of the difficulties of the Simple Client-Server Model by considerably expanding the role of the analysis/modelling system to

include the provision of viewing tools, macro programming language and other elements that are useful for the development of an ISDSS. By relying purely on the GIS for spatial data, the issue of user interface integration becomes irrelevant. However, the resulting system fails to capitalise on the distributed processing advantages of the Simple Client-Server Model.

Finally, in the Data Dispatch Model, the reliance on off-the-shelf software was dispensed with all together in favour of an entirely custom-built system. This object-oriented architecture can support ISDSS implementations that are very versatile and make effective use of distributed objects that are integrated tightly into the system. There were, however, two concerns. First, the implementation of the database management system as a remotely linked object could result in a data bottleneck, particularly where large volumes of information need to be exchanged (such as for the modelling objects). Secondly, building a large complex system using such an architecture is a considerable task, as even objects such as viewing tools would need to be implemented. Fortunately, class libraries for many common object types (including display forms) are widely available and so the system need not be developed entirely from scratch.

7.8 Summary

Several alternative architectures for building ISDSS have been proposed and an implementation of one of these was discussed. Through the application of a set of assessment criteria, it was found that each of the architectures have certain strengths and weaknesses. However, each of them addresses certain aspects of the problem in building flexible, sophisticated and powerful ISDSS. In particular, each of the architectures makes varying use of distributed computing techniques to spread the load associated with the processing intensive spatial modelling and analysis components. This is achieved through the deployment of distributed message passing techniques for performing dynamic data transfer between application components and between different applications.

Where the system also incorporated a GIS as part of the system (either for the display system or simply as a powerful spatial database), it was found that the architectural design of the system is limited considerably by the type and nature of the interapplication communication facilities provided by the GIS. The more sophisticated and 'complete' the IAC interface the better integrated the system could be, at both the data and control levels.

It was also found that many of these issues could be resolved by building systems that are completely self-contained and are not integrated 'live' with external applications. However, there are also problems associated with such systems, not least because the task

of building such systems is considerably more involved than for the other architectures discussed.

It was found that there are certain trade-offs to be made in constructing ISDSS – none of the architectures described is a perfect solution. In reality, the particular target application of an ISDSS and available resources will together determine which features are the most important and therefore the type of architecture that will be appropriate.

Chapter 8

Assessment

This chapter reviews and assesses the main findings of this research, covering the ISDSS taxonomy, the development and application of the GeoAnalyser system, and the architectures for intelligent spatial decision support. Each of these are also compared against possible alternatives and related work.

8.1 Target Review

The ultimate goal of this research has been to explore the fundamental issues involved in the integration of intelligent systems techniques with spatial decision support systems to form intelligent spatial decision support systems. In doing so a broad spectrum of theoretical and practical issues were considered, ranging from issues concerning performance and system architecture to the problem-solving capabilities of the resulting system.

This research has been founded on existing concepts of spatial decision support systems and a careful appreciation of how this class of system differs fundamentally from GIS. The ability of such systems for addressing ill or semi-structured spatial problems was exposed from the synthesis of the three main characteristics of SDSS: *interactive user interface*, *exploratory spatial analysis tools* and *explicit decision support features*. However, the limitations of SDSS were also recognised and it was shown that these result from the increasing complexity of modelling tasks and the inability of existing systems and modelling tools to make use of the huge volumes of spatial data that are now being collected.

It was shown that there are three key benefits of integrating intelligent systems techniques with SDSS: *enhancement of spatial modelling and spatial data analysis*, *knowledge processing* and *enhanced decision support*. These strengths arise from the symbolic and adaptive knowledge processing abilities of these techniques to learn, adapt, innovate, apply expert knowledge and create new knowledge. Furthermore, it was shown that the application of intelligent systems techniques for spatial decision-making falls into one of three categories: *data-oriented systems*, *model-oriented systems* and *decision-oriented systems*. The classification of existing systems demonstrated the effectiveness of this novel taxonomy.

Having established a theoretical framework for ISDSS, the research focused on the design and development issues arising from the construction of ISDSS. The various software engineering approaches taken with SDSS were used as a starting point. The numerous benefits of adopting a modular approach was described. In particular it was noted that several current SDSS utilise GIS as components within the overall systems, drawing on their superior display and spatial data handling capabilities. The analysis showed that a key ISDSS implementation issue concerns the specific objective for which an intelligent technique is used within the system. It was recognised that the various intelligent techniques possess unique spatial problem-solving properties and that there are benefits to be gained in combining techniques.

Issues concerning the integration of software components were examined in depth. Existing approaches to integrating components that include GIS, ranging from loosely coupled systems to tightly coupled systems, require a trade off between speed of data transfer, GIS-specificity and level of integration. It was found that there is a need for a method that permits high level data structures to be passed rapidly between coupled components. It was argued that interapplication communication techniques can be used to tightly link system components, permitting seamless joins to be made between components, while having the additional benefit of allowing the use of distributed processing resources. This *co-operative* approach to integration can take one of two forms: *indirect co-operation* or *direct co-operation* depending on whether the modelling system interacts directly with the display subsystem.

To assess the theoretical and practical framework developed for ISDSS, the GeoAnalyser system was developed as a prototype ISDSS. Designed as a generic system for developing and deploying spatial interaction models, GeoAnalyser combined a number of novel concepts. This system demonstrated the following aspects of ISDSS: integration of distributed system components using direct co-operative coupling based on Apple Events; provision of decision support facilities through an interactive user interface and spatial model; use of a genetic algorithm for model calibration; spatiotemporal data visualisation; and decision support tools capable of handling spatiotemporal data. GeoAnalyser also provided a plug-in architecture for deploying pre-built spatial models.

In order to test the effectiveness of these features, the system was applied to a complex spatial analysis task. GeoAnalyser was used to explore the spatiotemporal dynamics of high technology industry in South East England. The analysis provided the basis for developing a spatial interaction model of this industry. The model was coded as a plug-in for the server component of the system and calibrated using the genetic algorithm built into the client side. During the calibration, due to the tendency of the model to diverge, it was found necessary to partition the parameter space so that reasonable sets of solutions

were accessible. The results showed that the GA was then very useful for finding an optimal or near optimal set of parameters. The optimised parameters were tested against unseen data for the same industry sectors and locations. The best of these sets generated predictions with an average fit of 83%. This compares very favourably with the results of regression, which gave an average fit of 70%.

The operating characteristics of GeoAnalyser were also assessed using two different hardware and network set ups. Both showed clearly how the client-server architecture of GeoAnalyser can benefit the user. For model and GA related tasks, this translated in speed improvements of up to 95 times, as the user was able to make use of the superior processing power of the server machine. The assessment also clearly showed how the limitations of interapplication communication methods can impact the speed of display oriented tasks, which suffered degraded performance due to the low transmission rate (only 3-10 messages per second) of Apple Events, the technology used to link the client and server across the network.. This has significant implications for the design of such systems and methods of alleviating some of these difficulties were discussed.

The development of GeoAnalyser led to an assessment of the fundamental theoretical and practical issues regarding the employment of intelligent systems techniques in spatial data analysis and the construction of ISDSS. This foundation was used to develop several distinct theoretical architectures for the construction of ISDSS. These architectures were assessed in terms of ease and speed of information exchange, effectiveness of the user interface, degree of modularity (including how prebuilt components such as GIS can be integrated) and use of distributed computing resources.

In the following sections the main aspects of this research are compared with alternative methods.

8.2 ISDSS Taxonomy

The ISDSS taxonomy was developed to classify the application of intelligent techniques within the overall system. Three categories were proposed. Data-oriented systems employ intelligent techniques for the either data storage, query or conversion. Model-oriented systems use the techniques to model or analyse data. Decision-oriented systems use intelligent techniques for high level problem solving support. Each class of system was illustrated with reference to existing systems. In addition, the flexibility of the taxonomy was demonstrated by using it for the classification of systems that are not strictly ISDSS but nevertheless employ intelligent systems in their construction.

Though no direct alternatives have been proposed in the literature, several related taxonomies have been developed. Two of these are described below for comparison.

Alter describes a taxonomy of general decision support systems (Alter 1977) based on the degree to which the system's outputs can directly affect the decision outcome. Seven distinct types are proposed based on two broad classes: data-oriented systems (covering file drawer systems, data analysis systems and analysis information systems) and model-oriented systems (covering accounting models, representational models, optimisation models and suggestion models). While on the face of it the scheme appears to be compatible with that proposed here, two main criticisms can be made. First, the detailed scheme does not readily lend itself for the classification of intelligent systems, as at least once class (file drawer systems) will be empty if current ISDSS are considered. Secondly, the scheme cannot be used to classify systems employing intelligent systems for high level problem solving tasks, such as the use of a knowledge base for comparing decisions.

Focusing on spatial analysis, Goodchild presents a six class taxonomy of GIS spatial analysis operations (Goodchild 1987). Here operations are classified according to the types of data access required and the objects that are involved. Six classes of operation are proposed: (i) operations requiring access to only the attributes of one class of object; (ii) operations requiring access to attributes and locational information for a single class of objects; (iii) operations that create object-pairs from one or more class of object; (iv) operations which analyse attributes of object-pairs; (v) operations requiring access to attributes and locational information for more than once class of object; and (vi) operations which create a new class of object from an existing class. A classification such as this is not really designed to capture the general use of analysis techniques within an ISDSS. Spatial analysis operations executed within the framework of a GIS forms a fairly well defined but narrow class of operations compared to those that are routinely performed in a SDSS. Examples of operations that are not readily classified using this taxonomy include operations involving an expert system for enhancing the decision support features or operations employing fuzzy logic to modify the querying features of the spatial database.

To summarise, the ISDSS taxonomy proposed by this thesis is simple, provides a meaningful scheme for classifying the application of the essential defining component of the system, the intelligent systems technique, and can also be used for classifying systems that use these techniques but are not in themselves examples of ISDSS. Alternative taxonomies generally focus on aspects of spatial analysis, integration methods or are too broad to be of much use for this particular class of system.

8.3 Construction and Application of GeoAnalyser

The problems of integrating spatial analysis techniques with GIS and the development of SDSS that provide spatial analysis tools as well as geoprocessing tools has attracted a

great deal of interest in recent years. Several researchers have discussed the potential benefits of using distributed computing methods and hardware architectures (Dangermond 1988), for distributed geoprocessing and data management (McGregor 1988; Seaborn 1988; Abel, Kelly et al. 1994) as well as for providing the means to link system components together (Nyerges 1993). Some attempts have also been made to make use of networked workstations. Examples include the development of a distributed GIS environment (Cowen 1988), where some the spatial data manipulation and map generation functions of a standard central GIS are extended through more simplified interfaces on user machines, and the use of network-aware operating systems to distribute spatial tools among networked workstations (Ferreira and Menendez 1988). However there is generally very little work reported in the literature that makes serious attempts to use distributed computing for creating integrated tools for spatial decision support. Even fairly recent work on integrating spatial analysis with GIS fails to recognise the existence of other methods other than the usual dichotomy of loosely coupled and tightly coupled approaches (see for example Ding and Fotheringham 1992).

However, there are indications of wider interest in the GIS community of the shortcomings of GIS. A recent conference/workshop organised by the NCGIA discussed some of the interoperability and integration issues raised by this thesis as part of an effort to detail a global research agenda. It was felt that future GIS will be (Goodchild, Egenhofer et al. 1997):

- "distributed, with processing, storage of data, and user interaction occurring at locations that are potentially widely scattered; a user at location A might send data from location B to a server at location C to be processed, and have the results returned to location A for display;

- disaggregated, as monolithic systems developed by single vendors are replaced by plug and play components from different vendors that are designed to interoperate through conformity with industry-wide standards;

- decoupled, with the disaggregated components needed to complete a given complex of tasks being distributed over many networked systems;

- interoperable, a requirement that is clearly a precondition for all of the previous three."

From this conference, it is clear that there is now considerable interest in using IAC methods such as OLE/COM for both data integration (Hagehuelsmann 1997) and developing GIS as embeddable components (Gonçalves, Neves et al. 1997), designing distributed GIS (Koum 1997), making spatial data accessible over the Internet via

specialised servers (Vandenberg, Tuijman et al. 1997) and tightly coupling models with GIS using IAC methods (Karimi 1997).

Such research efforts are of great importance in the development of end user decision support systems that incorporate GIS within the overall decision support framework. In this context, the approach demonstrated in the construction of GeoAnalyser therefore provides valuable evidence of the effectiveness of distributed methods for both linking system components together and realising the true benefits of distributed data processing. This would have a particular benefit on highly computing intensive operations, reducing the need for less practical approaches, such as the use of supercomputers (Openshaw, Cross et al. 1990; Openshaw and Turton 1994)

The application of genetic algorithms for calibrating spatial interaction models is also new within the field of SDSS and GIS. However, GAs and similar techniques (e.g. genetic programming) have been utilised for other spatial problems, such as for the development of entirely new models (Openshaw and Turton 1994) and locational modelling (Stender 1994).

The issue of the temporal dimension in spatial data has long been recognised as a shortcoming of most SDSS and GIS (Robinson and Frank 1987). Acknowledging the necessity to offer temporal visualisation tools, several attempts have been made to use animation to view trends in spatial data over time (Dorling and Openshaw 1992). However this tool has not been adequately integrated within the system. Other work published in the literature uses external movie making programs that are not well integrated within the system's user interface. GeoAnalyser therefore demonstrates that animation can be performed within the system if image caching schemes are used to minimise redraw times.

As far as the application area is concerned, the work done in modelling high technology industry is really a preliminary investigation of the modelling potential of this industry. Further work is clearly needed in this area. Nevertheless, the use of GeoAnalyser in exploring spatiotemporal data in order to develop new spatial models and understanding demonstrates the value of the tools provided by the system.

8.4 Evaluation of ISDSS Architectures

Chapter 7 proposed four distinct architectures for ISDSS and compared theoretically the relative benefits and drawbacks of each of these. Of the four architectures, the client-server model was illustrated by a real system (GeoAnalyser), directly verifying the strengths and weaknesses of that particular architecture. A practical evaluation of the other three architectures is also desired but is beyond the scope of the work reported here.

Various architectures have been described in the literature in order to link spatial analysis with GIS and other forms of spatial visualisation tool for the development of SDSS. Two such architectures are discussed below for comparison.

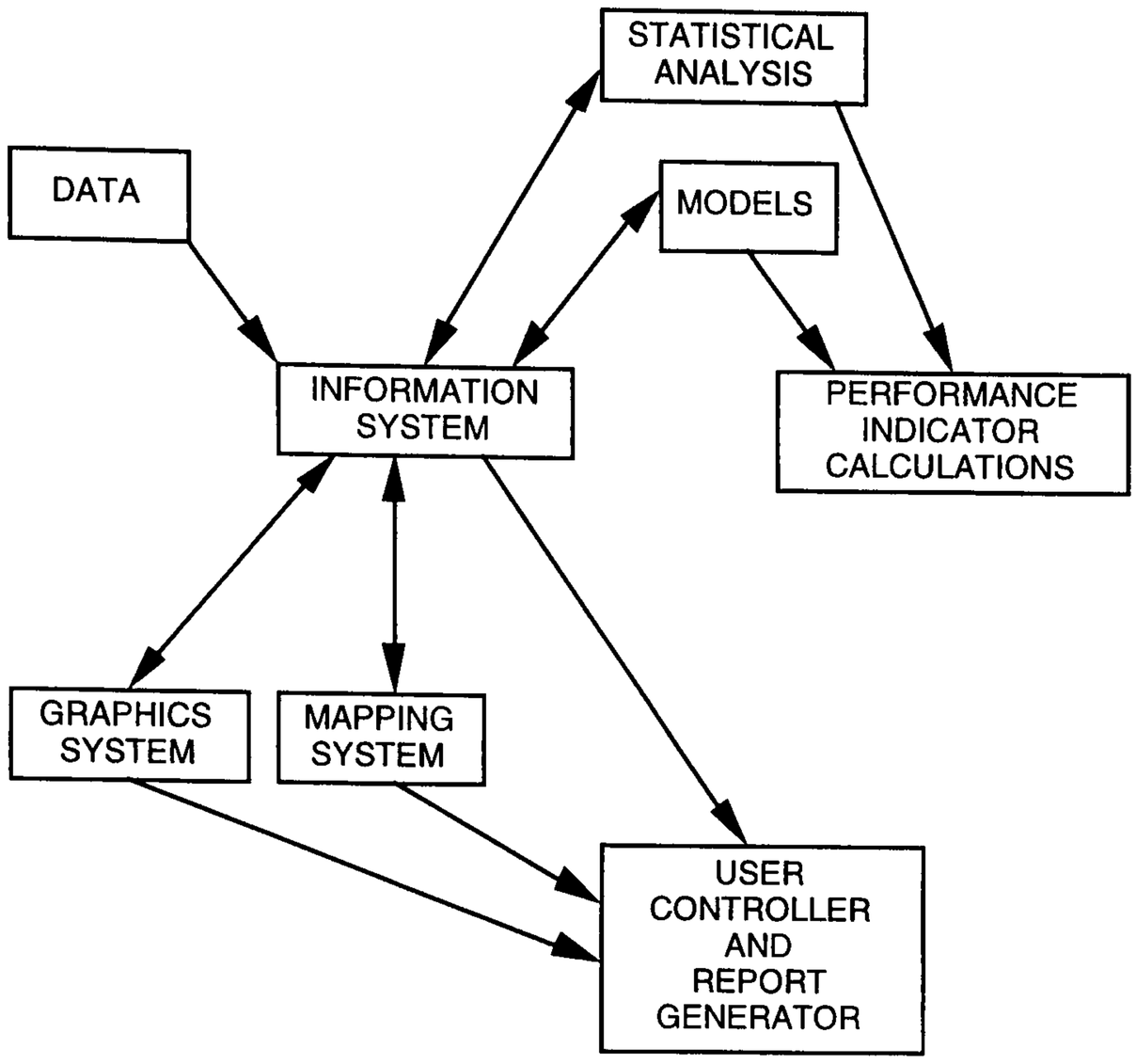


Figure 8.1 – Architecture for a Spatial Decision Support System (Birkin, Clarke et al. 1990)

Birkin et al describe an architecture for a fully integrated SDSS (Birkin, Clarke et al. 1990) which integrates mapping components, models, statistical methods, user interface and database into a single system for urban policy evaluation (see Figure 8.1). Architectures such as these suffer from several problems. First, they tend to be closed systems that cannot easily be adapted for different applications. Secondly, the architectures are designed purely for standalone computers and are therefore incapable of utilising the benefits of distributed processing. Thirdly, these architectures exhibit little reuse of existing systems (for example, in the system described by Birkin et al, with the exception of the database the entire system is written from scratch). Finally, and perhaps more importantly, the architectures do not lend themselves readily to the integration of advanced exploratory analysis methods, including intelligent systems techniques, as they are not designed to accommodate a high level of interaction between system components.

Armstrong et al describe the development of a prototype SDSS for the analysis of location-allocation problems (Armstrong, Rushton et al. 1991). The architecture for this system revolves around a central database, which is used to provide data to the analytical modules and store the results of analysis so that these may be displayed using mapping and report generators (in this the Atlas GIS was used for displaying maps). What is interesting about this early SDSS is that the analytical modules and mapping/report

modules were run on separate computers, which were able to transfer data via a parallel port connection system. While quite a crude set-up, this allowed decision makers to achieve a far more rapid turnaround in being to view model results as each model run typically took about 10 minutes. Thus this system illustrates the same speed advantages of a distributed system that were observed in GeoAnalyser. However, as the two parts of the system were loosely coupled together, the level of interaction is much lower than that achieved with GeoAnalyser.

The advancements made in networking technologies have permitted the development of distributed software such as GeoAnalyser, where the user is largely unaware of which computer the system is running on. Recent advances in software technologies offer many benefits for the construction of useful SDSS and ISDSS. In the last few years, particularly with the inception of the Java technology in the last two years, commercial systems have embraced object oriented design, network computing and distributed objects. Two-tier systems (i.e. client-server systems) are now widespread. Many system developers are now focusing on three-tier systems, where the core operating logic of the system is encapsulated in the middle tier while the user interface and database reside on separate tiers. More generally there are clear signs of the adoption of n-tier systems, with business systems now being constructed using application servers and other forms of middleware.

The Enterprise Java Beans component model is another important technology which, it is hoped, will lead to the adoption of plug-and-play server software, that can be rapidly constructed using prebuilt components.

If such technologies are to be similarly adopted by the developers of SDSS, competent workable architectures for the utilisation of distributed systems need to be developed. The architectures proposed here therefore provide a foundation for moving in this direction.

8.5 Summary

The main aims of this research have been to establish a theoretical and practical basis for the development of intelligent spatial decision support systems. The contributions of this research have been to develop a taxonomy of the intelligent systems techniques applied to spatial problem-solving, the design and implementation of a demonstration ISDSS and its application for the development of a spatial model of high technology industry in South East England, and an analysis of viable generic architectures proposed for ISDSS.

The ISDSS taxonomy was compared with taxonomies for general decision support systems and GIS spatial analysis operations and it was found that both of these do not properly classify the use of complex analysis techniques within ISDSS.

GeoAnalyser was compared with similar SDSS that have been developed and it was found that very few systems utilise distributed computers at all, let alone as part of a tightly integrated system. Also, the spatiotemporal visualisation tools are an advancement on similar reported work, where the animation functions operate independently of the main system.

Finally, the ISDSS architectures were compared with two SDSS, one of which consisted of loosely coupled systems running on two linked machines. It was also noted that, while commercial systems have begun to use distributed computing techniques in the past few years, similar developments have not been made in SDSS.

Chapter 9

Conclusions and Future Work

This chapter presents the conclusions of this thesis. Further developments of the GeoAnalyser system, ISDSS architectures and ISDSS are also discussed.

9.1 Conclusions

The main conclusions of this thesis are presented below on a chapter by chapter basis.

Building on the survey of spatial problem-solving and spatial decision support systems given in Chapter 2, Chapter 3 provides an analysis of the arguments for the development of ISDSS. It reveals the main implementation routes available for their development, concentrating on integration issues. The conclusions that are drawn from this are:

- Intelligent systems techniques each have unique spatial problem-solving characteristics. As a result, a combination of these techniques and traditional spatial analysis methods can overcome the limitations of any one technique.

- Existing SDSS that are developed as integrated systems use either loose coupling or tight coupling to bind together the various system modules. Both of these forms of coupling involve a trade-off between speed of data transfer, GIS-specificity and the level of integration achieved.

- ISDSS may be effectively classified in terms of three categories: data-oriented systems, model-oriented systems and decision-oriented systems. These categories clearly describe the architectural and functional role of intelligent techniques within the ISDSS.

- The integration of GIS with spatial analysis techniques and spatial models is a key issue in the construction of ISDSS.

Chapter 4 describes the design and implementation of the GeoAnalyser ISDSS are described. The main design goals of this system were: provision of spatiotemporal visualisation tools and tools for developing, calibrating and testing spatial interaction models; incorporates support for temporal data while maintaining fast access to spatial data structures; and exploitation of distributed computing resources. From this it can be concluded that:

- Modular system design approaches are highly effective for building distributed ISDSS. These approaches offer flexible choice of programming language and development environment, design autonomy and parallel execution.

- Plug-in technologies are an important means of introducing extensibility into an ISDSS. This is particularly relevant for models, as in GeoAnalyser, where plug-ins allow different spatial models to be made available within the system.

- Where relatively small amounts of data are involved, simple in-memory spatiotemporal data structures offer virtually instantaneous access and help to simplify the design of the system. GeoAnalyser uses a list-type data structure to represent a spatiotemporal dataset as a linear sequence of linked spatial frames, each with its own time stamp.

- For real-time animation to be integrated with the system, mapping subsystems need to be streamlined and use made of optimisation techniques to reduce redraw times.

In Chapter 5, the GeoAnalyser ISDSS is used for the real world problem of building a spatial interaction model for high technology industry in the South East of England. The system was initially used to explore the dynamics of the industry. A prototype plug-in model was developed from this analysis and the model calibrated using the built-in genetic algorithm. The performance characteristics of GeoAnalyser were measured for two different hardware and network configurations. It can be concluded that:

- For the UK at least, the lack of suitable data on high technology industry severely restricts the scope of modelling that can be achieved.

- In conjunction with other more static visualisation techniques, animation techniques are a useful method of visualising the temporal dynamics of spatial data. The techniques can also be successfully integrated within the user interface of the SDSS.

- Genetic algorithms are highly effective for calibrating complex spatial interaction models provided that the parameter space is suitably partitioned to prevent the individual population being overrun by divergent solutions.

- Interapplication communication techniques can be successfully used to integrate system components, achieving a tight yet flexible coupling that permits linked operations, offers fast data transfer between components and delivers the performance benefits of components that are distributed over a network.

- The performance of display-oriented operations is highly sensitive to the performance characteristics of the interapplication communication technique being used to link components. System responsiveness can be improved using a caching scheme to reduce data communications overhead, as display-oriented operations do not generally alter the underlying data.

Chapter 6 assesses the user interface and implementation issues arising from the development of GeoAnalyser and its application in real world problem-solving. Generalising on these findings it then assesses the main implementation issues concerning the integration of intelligent systems techniques. It concludes that:

- Client-server techniques help to reduce considerably the computational overhead of complex spatial modelling on the user's machine. These techniques are therefore a valuable means of making effective use of distributed computing resources and multiprocessor machines.

- Display-oriented operations that involve client-server communication depend critically on the speed of communication between the client and server applications.

- Access to powerful analysis techniques (such as genetic algorithms in the case of GeoAnalyser) and visualisation tools through a flexible user interface is essential for the successful exploration of spatial data using an ISDSS.

- Spatial model development and spatial model deployment are not necessarily exclusive tasks and can be accomplished successfully within the same application.

- Intelligent systems techniques are distinct from other modelling methods and techniques typically used in spatial analysis. The most important consideration for the design of the overall system is that the former operate in two distinct phases, consisting of the development and application phases. Intelligent techniques require different levels of control and interaction for these two phases. Consequently, each phase has its own distinct set of user interface requirements that will have a direct bearing on the overall user interface and system design.

- Key implementation issues for developing ISDSS are the incorporation of a decision support framework, support for exploratory spatial analysis, integration of intelligent systems techniques and integration of distributed system components.

Chapter 7 presents and assesses novel architectures for building ISDSS using distributed components. The assessment criteria are: level of user interface integration, data integration and problem-solving capability. It is concluded that:

- GIS can be used for the main display component of an ISDSS with scope for building well integrated systems provided suitable interapplication interfaces (for the display, storage and retrieval of data) are either already available within the GIS or can be built on top (using either macros or external executable modules).

- Three tier ISDSS architectures and higher are better able to provide high level decision support, model base management and other management functions than simple client-server systems. The middle layer can be used to encapsulate key application details in a manner analogous to the way that business rules are stored in commercial multi-tier systems.

- For more complex ISDSS architectures there is a trade off between the desired level of functionality, the benefits of using distributing components and the level of component reuse that can be achieved.

- Overall it was concluded that there is no single best architecture for ISDSS and that the particular design adopted by an application developer will depend on the specific requirements of the problem domain together with user expectations.

Chapter 8 assesses the ISDSS taxonomy, development and application of GeoAnalyser and ISDSS architectures. The conclusions of this analysis are:

- The ISDSS taxonomy provides a simple and effective method of classifying how intelligent systems are utilised within the overall system. The taxonomy is general enough to be applicable to other systems that use intelligent systems but are not in themselves ISDSS.

- GeoAnalyser offers an advancement over existing systems, particularly in its use of distributed processing to link system components together, integrated spatiotemporal visualisation techniques and genetic algorithms for calibrating spatial interaction models. Also the work done on modelling high technology industry provides a foundation for future work in this area.

- The ISDSS architectures proposed are novel in the area of SDSS and important as they offer a useful platform on which to reap the benefits of recent advances in Internet and communications technology.

9.2 Future Work

There are many possibilities for extending this research work. Some of these possibilities are discussed below.

The most immediate extension to this work would be to refine and develop further the GeoAnalyser system. As a research prototype this system has proved the viability of its client-server architecture. The application could be refined further through the introduction of a transport model, richer display tools, improved data handling support and the addition of high level problem-solving support (which would necessitate a reworking of the internal data structures used). Most of these refinements could be made incrementally, owing to the object-oriented design of the system. Due to the generic nature of the system, another useful extension to this work would be to build an application programming interface for plug-in model development. This would permit additional spatial models to be developed by either the end user or third parties.

Current interest in interoperable GIS will lead to the eventual enhancement of the IAC facilities provided by GIS. This will offer developers the opportunity to build sophisticated ISDSS from prebuilt subsystems.

Further work could also be undertaken to verify the various architectures proposed in this thesis, some of which use GIS as a key component. This would prove the validity of the architectures and would, it is believed, result in useful spatial analysis tools.

In Chapter 3 the complementary nature of intelligent systems was noted. This complementarity has led to the widespread development of hybrid systems incorporating the benefits of each of the techniques while easing the limitation of any one technique. There is however considerable scope for researching the application of hybrid systems in spatial analysis and decision support.

Ultimately it is hoped that this research will pave the way for the development of general architectures for spatial decision support incorporating intelligent systems techniques as a standard component. Such systems might be generalised in one of two ways. First, generic applications could be built that provide a range of facilities for the decision maker, rather like the way a spreadsheet application offers a whole range of facilities for numerical calculation. Such applications would clearly entail huge development expense in order to build in a broad range of features that would make the application suitable for a wide spectrum of uses. Secondly, application builders or toolkits that permit the construction of specialised applications could be developed. In both cases, this research has demonstrated the value of distributed architectures, extensible applications and

spatiotemporal visualisation. Further work in this area is now being undertaken to explore some of these possibilities (Sikder 1998).

Another important consideration is the increasing use of the Java programming language, related Java technologies (such as Remote Method Invocation for message passing between remote objects) and CORBA. These Internet-oriented, cross-platform technologies are being applied commercially for the development of many novel forms of applications that harness the ability to share data and applications globally (such as groupware and workflow systems). One possibility is to build distributed applications that support collaborative decision making. This could clearly have enormous implications for planning.

While this research has focused on the use of local area networks for linking the various components within an ISDSS, technologies are now in place to successfully make use of the Internet for this task. Such possibilities could have a profound effect on the very nature of spatial problem-solving, presenting opportunities for applications that run on fast servers but are accessible globally and offer sophisticated decision support in a collaborative environment.

References

Abel, D. J., P. J. Kelly, et al. (1994), "The systems integration problem," *International Journal of Geographical Information Systems* **8**(1): pp. 1-12.

Abel, D. J., S. K. Yap, et al. (1992), "Environmental decision support system project: an exploration of alternative architectures for geographical information systems," *International Journal of Geographical Information Systems* **6**(3): pp. 193-204.

Abercrombie, P. (1944), *Greater London Plan 1944*, HMSO.

Aleksander, I. and H. Morton (1990), *Introduction to Neural Computing*, North Oxford Press.

Allen, P. M. and M. Sanglier (1979), "A dynamic model of growth in a central place system," *Geographical Analysis* **11**(3): pp. 256-272.

Allen, P. M. and M. Sanglier (1981), "Urban evolution, self-organisation, and decisionmaking," *Environment and Planning A* **13**: pp. 167-183.

Alter, S. (1977), "A taxonomy of decision support systems," *Sloan Management Review* **19**(39): pp. 39-56.

Altman, D. (1994), "Fuzzy set theoretic approaches for handling imprecision in spatial analysis," *International Journal of Geographic Information Systems* **8**(3): pp. 271-289.

Apple Computer (1993), *Inside Macintosh: Interapplication Communication*, Addison Wesley.

Armstrong, M. P., S. De, et al. (1990), "A knowledge-based approach for supporting locational decisionmaking," *Environment and Planning B* **17**: pp. 341-364.

Armstrong, M. P. and P. J. Densham (1990), "Database organisation strategies for spatial decision support systems," *International Journal of Geographical Information Systems* **4**(1): pp. 3-20.

Armstrong, M. P., G. Rushton, et al. (1991), "Decision support system for regionalization: A spatial decision support system for regionalizing service delivery times," *Computers, Environment and Urban Systems* **15**: pp. 37-53.

Arthur, W. B. (1989), *"Silicon Valley" locational clusters: when do increasing returns imply monopoly*, Working Paper 89-007, Santa Fe Institute.

Banai-Kashani, A. R. (1990), "Dealing with uncertainty and fuzziness in development planning: a simulation of high-technology industrial location decisionmaking by the analytic hierarchy process," *Environment and Planning A* **22**: pp. 1183-1203.

Begg, I. C. and G. C. Cameron (1988), "High Technology Location and the Urban Areas of Great Britan," *Urban Studies* **25**: pp. 361-379.

Berson, A. (1992), *Client-server architecture*, McGraw-Hill.

Bill, R. and D. Fritsch (1991), *Grundlagen der Geoinformationssysteme. Band 1 Hardware, Software und Daten.*, Karlsruhe, Germany, Wichmann Verlag.

Birkin, M., G. Clarke, et al. (1990), Elements of a model-based geographical information system for the evaluation of urban policy, *Geographical Information Systems: Developments and Applications* Ed. L. Worrall, London, Belhaven Press, pp. 132-162.

Blum, E. (1972), "Deployment Research of the New York City Fire Project," *Urban Analysis* **1**: pp. 63-94.

Booker, A. and J. Pryor (1994), *The Economy of Hertfordshire*, Hertfordshire County Council.

Bracken, I. and C. Webster (1989), "Towards a typology of geographical information systems," *International Journal of Geographical Information Systems* **3**(2): pp. 137-152.

Bradstock, P. (1994), Director of Oxford Trust, *Personal communication.* 17 October 1994.

Breheny, M. J. and R. McQuaid (1987), H.T.U.K. The development of the United Kingdom's major centre of high technology industry, *The Development of High Technology Industries* Eds. M. J. Breheny and R. McQuaid, London, Croom Helm, pp. 296-354.

Britton, J. (1987), High Tech Industry in Canada, *The Development of High Technology Industries* Eds. M. Breheny and R. McQuaid, Croom Helm.

Burrough, P. A. (1986), *Five reasons why geographical information systems are not being used efficiently for land resources assessment*, Proceedings of Auto Carto, London, UK, Auto Carto London Ltd, Royal Institute of Chartered Surveyors, pp. 139-148.

Burrough, P. A. (1986), *Principles of Geographical Information Systems for Land Resource Assessment*, Oxford, Clarendon.

Burrough, P. A. (1989), "Fuzzy mathematical methods for soil survey and land evaluation," *Journal of Soil Science* **40**: pp. 477-492.

Burrough, P. A. (1990), "Preface - Methods of spatial analysis in GIS," *International Journal of Geographical Information Systems* **4**(3): pp. 221-223.

Burrough, P. A. (1992), "Development of intelligent geographical information systems," *International Journal of Geographical Information Systems* **6**(1): pp. 1-11.

Butchart, R. L. (1987), "A new UK definition of the high technology industries," *Economic Trends* **400**(February): pp. 82-88.

Carlson, E., J. Bennett, et al. (1974), *The Design and Evaluation of an Interactive Geo-Data Analysis and Display System*, Information Processing 74. The Proceedings of the IFIP Congress 1974, North Holland, New York, pp. 1057-61.

Carver, S. J. (1991), "Integrating multi-criteria evaluation with GIS," *International Journal of Geographical Information Systems* **5**(3): pp. 321-339.

Clark, J. D. (1992), *Windows Programmer's Guide to OLE/DDE*, SAMS/Prentice Hall Computer Publishing.

Codd, E. F. (1982), "Relational database: a practical foundation for productivity," *Communications of the ACM* **25**(2): pp. 109-117.

Cowen, D. J. (1988), "GIS versus CAD versus DBMS: what are the differences?," *Photogrammetric Engineering and Remote Sensing* **54**: p. 1551.

Cowen, D. J. (1988), *A Hypercard based workstation for a distributed GIS network*, Proceedings GIS/LIS '88, pp. 285-294.

Dangermond, J. (1988), *A technical architecture for GIS*, Proceedings GIS/LIS '88, pp. 561-570.

Davis, J. R., P. T. Compagnoni, et al. (1987), "Roles for knowledge-based systems in environmental planning," *Environment and Planning B* **14**: pp. 239-254.

Davis, J. R. and I. W. Grant (1987), "ADAPT: a knowledge-based decision support system for producing zoning schemes," *Environment and Planning B* **14**: pp. 53-66.

Davis, L. (1991), *Handbook of genetic algorithms*, New York, Van Nostrand Reinhold.

Dekker, L. (1995), *QGAME Genetic Algorithm Development Kit*, Technical Report, Department of Computer Science, University College London.

Densham, P. and G. Rushton (1988), Decision support systems for locational planning, *Behavioural Modelling in Geography and Planning* Eds. R. G. Golledge and H. Timmermans, London, Croom Helm, pp. 56-90.

Densham, P. J. (1994), "Integrating GIS and Spatial Modelling: Visual Interactive Modelling and Location Selection," *Geographical Systems* **1**: pp. 203-219.

Department of Education (1993), *HEFCE Funded Institutions*,

Diamond, J. T. and J. R. Wright (1988), "Design of an integrated spatial information system for multiobjective land-use planning," *Environment and Planning B* **15**: pp. 205-214.

Ding, Y. and A. S. Fotheringham (1992), "The integration of spatial analysis and GIS," *Computers, Environment and Urban Systems* **16**: pp. 3-19.

Dorling, D. and S. Openshaw (1992), "Using computer animation to visualize space-time patterns," *Environment and Planning B* **19**: pp. 639-650.

Engelen, G. (1988), "The theory of self-organization and modelling complex urban systems," *European Journal of Operational Research* **37**: pp. 42-57.

Evans, S. and J. Norback (1985), "The Impact of a Decision Support System for Vehicle Routeing in a Foodservice Supply Situation," *Journal of the Operational Research Society* **36**: pp. 467-72.

Faini, R. (1984), "Increasing returns, non-traded inputs and regional development," *The Economic Journal* **94**: pp. 308-323.

Fedra, K. (1993), GIS and Environmental Modelling, *Environmental Modelling with GIS* Eds. M. F. Goodchild, B. O. Parks and L. T. Steyaert, Oxford University Press, pp. 35-50.

Feldman, M. M. A. (1984), Biotechnology and local economic growth: The American pattern, *Silicon Landscapes* Eds. P. Hall and A. Markusen, Boston, Mass., Allen & Unwin, pp. 65-79.

Ferreira, J. and A. Menendez (1988), *Distributing spatial analysis tools among networked workstations*, URISA '88 - Mapping the Future, Washington DC, Urban and Regional Information Systems Association, pp. 200-215.

Fischer, M. and P. Nijkamp, Eds. (1993), *Geographic Information Systems, Spatial Modelling, and Policy Evaluation*, Berlin, Springer Verlag.

Forsyth, R. (1986), *Machine Learning: Applications in expert systems and information retrieval*, Ellis Horwood Ltd.

Geoffrion, A. M. (1983), "Can OR/MS evolve fast enough?," *Interfaces* **13**: p. 10.

Gillespie, A., J. Howells, et al. (1987), IT Production Industries in Europe, *The Development of High Technology Industries* Eds. M. Breheny and R. McQuaid, Croom Helm.

Goldberg, D. E. (1989), *Genetic algorithms in search, optimisation and machine learning*, Addison-Wesley.

Goldberg, M., D. G. Goodenough, et al. (1985), "A hierarchical expert system for updating forestry maps with Landstat data," *Proceedings of the IEEE* **73**(6): pp. 1054-1063.

Gonçalves, P. P., N. Neves, et al. (1997), *Interoperability of Geographic Information: From the spreadsheet to virtual environments*, International Conference and Workshop on Interoperating Geographic Information Systems, 3-6 December 1997, Santa Barbara, CA, NCGIA, WWW (http://www.ncgia.ucsb.edu/conf/interop97/).

Goodchild, M. and K. Kemp, Eds. (1990), *Introduction to GIS*, Santa Barbara, CA, National Center for Geographic Information and Analysis, University of California.

Goodchild, M. F. (1987), "A spatial analytical perspective on geographical information systems," *International Journal of Geographical Information Systems* **1**(4): pp. 327-334.

Goodchild, M. F. (1987), *Towards an enumeration of and classification of GIS functions*, Proceedings, IGIS 87: the Research Agenda, Washington DC: NASA, pp. 67-77.

Goodchild, M. F. (1991), *Progress on the GIS research agenda*, Proceedings EGIS '91, pp. 342-350.

Goodchild, M. F., M. J. Egenhofer, et al. (1997), *Interoperating GISs - Report of a Specialist Meeting Held under the Auspices of the Varenius Project, Panel on Computational Implementations of Geographic Concepts*, NCGIA, University of California, Santa Barbara, California.

Goodchild, M. F. and V. T. Noronha (1987), Location-allocation and impulsive shopping: The case of gasoline retailing, *Spatial Analysis and Location-Allocation Models* Eds. A. Ghosh and G. Rushton, New York, Van Nostrand Reinhold, pp. 121-136.

Goodenough, D. G., M. Goldberg, et al. (1987), "An expert system for remote sensing," *IEEE Transactions on Geoscience and Remote Sensing* **GE-25**(3): pp. 349-359.

Goonatilake, S. and S. Khebbal (1995), *Intelligent hybrid systems*, John Wiley & Sons.

Guariso, G. and H. Werthner (1989), *Environmental Decision Support Systems*, Chichester, Sussex, Ellis Horwood.

Guevara, J. A. and D. Bishop (1985), *A fuzzy and heuristic approach to segment intersection detection and reporting*, Seventh International Symposium on Computer-assisted Cartography, Washington DC, p. 228.

Hagehuelsmann, A. (1997), *Hot Links as a new Way of Data Integration in a Distributed Computing Environment*, International Conference and Workshop on Interoperating Geographic Information Systems, 3-6 December 1997, Santa Barbara, CA, NCGIA, WWW (http://www.ncgia.ucsb.edu/conf/interop97/).

Hall, P., M. Breheny, et al. (1987), *Western Sunrise: The genesis and growth of Britain's major high tech corridor*, London, Allen & Unwin.

Han, S.-Y. and T. J. Kim (1989), "An application of expert systems in urban planning: site selection and analysis," *Computers, Environment and Urban Systems* **13**: pp. 243-254.

Han, S.-Y., T. J. Kim, et al. (1991), "XPLanner: A knowledge-based decision support system for facility management and planning," *Environment and Planning B* **18**: pp. 205-224.

Henderson, J. and A. Scott (1987), The American Semiconductor Industry, *The Development of High Technology Industries* Eds. M. Breheny and R. McQuaid, Croom Helm.

Heuvelink, G. B. M., P. A. Burrough, et al. (1989), "Propagation of errors in spatial modelling with GIS," *International Journal of Geographical Information Systems* **3**(4): pp. 303-322.

Holland, J. H. (1975), *Adaptation in natural and artificial systems*, Ann Arbor, The University of Michigan Press.

Hopkins, L. D. (1984), "Evaluation of methods for exploring ill-defined problems," *Environment and Planning B* **11**: pp. 339-348.

Hopkins, L. D. and M. P. Armstrong (1985), *Analytic and cartographic data storage: a two-tiered approach to spatial decision support systems*, Seventh International Symposium on Computer-assisted Cartography, Washington DC, pp. 430-439.

Isard, W. (1956), *Location and space-economy*, New York, Technology Press of MIT.

Jackson, P. (1986), *Introduction to Expert Systems*, Addison-Wesley.

Karimi, H. A. (1997), *Interoperable GIS Applications: Tightly Coupling Environmental Models with GISs*, International Conference and Workshop on Interoperating Geographic Information Systems, 3-6 December 1997, Santa Barbara, CA, NCGIA, WWW (http://www.ncgia.ucsb.edu/conf/interop97/).

Kessel, S. R. (1990), "An Australian geographical information and modelling system for natural area management," *International Journal of Geographical Information Systems* **4**(3): pp. 333-362.

Kohsaka, H. (1993), "A Monitoring and Locational Decision Support System for Retail Activity," *Environment and Planning A* **25**: pp. 197-211.

Koum, G. (1997), *Structural Design of Distributed Geographics Information Systems Based on a High Order Logic*, International Conference and Workshop on Interoperating Geographic Information Systems, 3-6 December 1997, Santa Barbara, CA, NCGIA, WWW (http://www.ncgia.ucsb.edu/conf/interop97/).

Lawton Smith, H. (1991), "The role of incubators in local industrial development: the cryogenics industry in Oxfordshire," *Entrepreneurship & Regional Development* **3**: pp. 175-194.

Leung, Y. (1982), "Approximate characterisation of some fundamental concepts of spatial analysis," *Geographical Analysis* **14**(1): pp. 29-40.

Liddiard, G. (1994), Buckinghamshire County Council, *Personal communication*. 13 October 1994.

Lösch, A. (1954), *The Economics of Location*, New Haven, Conn., Yale University Press.

MacDonald, S. (1983), "High technology policy and the Silicon Valley model," *Prometheus* **1**(2): pp. 253-265.

Maguire, D. J. (1991), An Overview and Definition of GIS, *Geographical Information Systems* Eds. D. J. Maguire, M. F. Goodchild and D. W. Rhind, Essex, England, Longman Scientific and Technical, pp. 9-20.

Markusen, A. R. (1984), High tech jobs, markets and economic development prospects: evidence from California, *Silicon Landscapes* Eds. P. Hall and A. Markusen, Boston, Mass., Allen & Unwin, pp. 134-143.

McBratney, A. B. and A. W. Moore (1985), "Application of fuzzy sets to climatic classification," *Agricultural and Forest Meteorology* **35**: pp. 165-185.

McGregor, D. (1988), *Geographic information system trends*, Proceedings GIS/LIS '88, pp. 915-921.

Micro Database Systems Inc. (1987), *GURU Reference Manual*, Micro Database Systems Inc., PO Box 248, Lafayette, IN 47902. USA.

NCGIA (1989), "The research plan of the National Center for Geographic Information and Analysis," *International Journal of Geographical Information Systems* **3**(2): pp. 117-136.

Nishioka, H. and A. Takeuchi (1987), The Development of High Tech Industry in Japan, *The Development of High Technology Industries* Eds. M. Breheny and R. McQuaid, Croom Helm.

Nyerges, T. L. (1993), Understanding the Scope of GIS: Its Relationship to Environmental Modelling, *Environmental Modelling with GIS* Eds. M. F. Goodchild, B. O. Parks and L. T. Steyaert, Oxford University Press, pp. 75-93.

Oakey, R. (1984), High-technology industries and agglomeration economies, *Silicon Landscapes* Eds. P. Hall and A. Markusen, Boston, Mass., Allen & Unwin, pp. 94-117.

Openshaw, S. (1988), "Building an Automated Modeling System to Explore a Universe of Spatial Interaction Models," *Geographical Analysis* **20**(1): pp. 31-46.

Openshaw, S. (1991), "A view on the GIS crisis in geography, or, using GIS to put Humpty-Dumpty back together again," *Environment and Planning A* **23**: pp. 621-628.

Openshaw, S. (1994), "Computational human geography: towards a research agenda," *Environment and Planning A* **26**: pp. 499-508.

Openshaw, S. (1994), Neuroclassification of spatial data, *Neural Nets: applications in Geography* Eds. B. C. Hewitson and R. G. Crane, Boston, Kluwer Academic Publishers, pp. 53-70.

Openshaw, S. (1995), "Human systems modelling as a new grand challenge in science - what has happened to the science in social science," *Environment and Planning A* **27**: pp. 159-164.

Openshaw, S., M. Charlton, et al. (1987), "A Mark 1 Geographical Analysis Machine for the automated analysis of point data sets," *International Journal of Geographical Information Systems* **1**: pp. 335-358.

Openshaw, S., A. Cross, et al. (1990), "Building a prototype Geographical Correlates Exploration Machine," *International Journal of Geographical Information Systems* **4**(3): pp. 297-311.

Openshaw, S. and I. Turton (1994), *Building New Spatial Interaction Models Using Genetic Programming*, in T. C. Fogarty, editor, Evolutionary Computing, Lecture Notes in Computer Science, Leeds, UK, 11-13 April 1994. Springer-Verlag.

Openshaw, S. and I. Turton (1994), *Building new spatial interactive models using genetic programming*, Technical Report, School of Geography, University of Leeds, Leeds, UK.

Openshaw, S. and C. Wymer (1991), A neural net classifier system for handling census data, *Neural Networks for Statistical and Economic Data* Ed. F. Murtagh, Dublin, Munotec, pp. 73-86.

Pascoe, R. T. and J. P. Penny (1990), "Construction of interfaces for the exchange of geographic data," *International Journal of Geographic Information Systems* **4**(2): pp. 147-156.

Peters, E. E. (1991), *Chaos and Order in the Capital Markets*, London, John Wiley & Sons.

Pottier, C. (1987), Location of High Tech Industries in France, *The Development of High Technology Industries* Eds. M. Breheny and R. McQuaid, Croom Helm.

Prigogine, I. and I. Stengers (1984), *Order out of chaos*, New York, Bantam Books.

Quinlan, J. R. (1979), Discovering rules by induction from large collections of examples, *Expert Systems in the Micro Electronic Age* Ed. D. Mitchie, Edinburgh, Edinburgh University Press, pp. 168-201.

Ribeiro-Filho, J. and P. Treleaven (1994), "Genetic Algorithms Programming Environments," *IEEE Computer* **27**(6): pp. 28-43.

Robinson, G. and M. Jackson (1985), *Expert systems in map design*, Seventh International Symposium on Computer-assisted Cartography, Washington DC, pp. 430-439.

Robinson, V. (1983), *Urban Data Management Software (U.D.M.S.) Package - User's Manual*, United Nations Centre for Human Settlements (HABITAT), Nairobi.

Robinson, V. B. (1988), "Some implications of fuzzy set theory applied to geographic databases," *Computers, Environment and Urban Systems* **12**: pp. 89-97.

Robinson, V. B. and A. U. Frank (1987), "Expert systems for geographic information systems," *Photogrammetric Engineering and Remote Sensing* **53**(10): pp. 1435-1441.

Sandhu, R. S. and P. Treleaven (1995), *Intelligent Geographical Information Systems*, Proceedings GISRUK'95, Newcastle upon Tyne, England, UK, pp. 107-111.

Sandhu, R. S. and P. Treleaven (1996), *Client-Server Approaches to Model Integration within GIS*, Third International Conference/Workshop on Integrating GIS and Environmental Modelling, 21-25 January 1996, Santa Fe, New Mexico, NCGIA Publications Office, CD.

Saxenian, A. (1983), "The Genesis of Silicon Valley," *Built Environment* **9**(1): pp. 7-17.

Sayer, A. and K. Morgan (1987), Electronics, *The Development of High Technology Industries* Eds. M. Breheny and R. McQuaid, Croom Helm.

Seaborn, D. W. (1988), *Distributed processing and distributed databases in GIS - separating hype from reality*, Proceedings GIS/LIS '88, pp. 141-144.

Sikder, I. U. (1998), *Personal communication*. 26 January 1998.

Smith, T. R. (1984), "Artificial Intelligence and its Applicability to Geographical Problem Solving," *The Professional Geographer* **36**(2): pp. 147-158.

Sommerville, I. (1989), *Software Engineering*, Addison-Wesley.

Star, J. and J. Estes (1990), *Geographic Information Systems: an introduction*, Englewood Cliffs, NJ, Prentice Hall.

Stender, J. (1994), Standort 2000: Locational modelling of the Brandenburg area using genetic algorithms, *Genetic Algorithms in Optimisation, Simulation and Modelling* Eds. J. Stender, E. Hillebrand and J. Kingdon, IOS Press, pp. 219-259.

Stott, M. (1994), Oxfordshire County Council, *Personal communication*. 11 October 1994.

Sui, D. Z. (1992), "A fuzzy GIS modeling approach for urban land evaluation," *Computers, Environment and Urban Systems* **16**: pp. 101-115.

Taylor, T. (1984), High-technology industry and the development of science parks, *Silicon Landscapes* Eds. P. Hall and A. Markusen, Boston, Mass., Allen & Unwin, pp. 134-143.

Thompson, J. M. Y. and H. B. Stewart (1986), *Nonlinear Dynamics and Chaos*, John Wiley & Sons.

Treleaven, P. and S. Goonatilake (1992), *Intelligent Financial Technologies*, Workshop on Parallel Problem Solving from Nature: Applications in Statistics and Economics, Statistical Office of the European Communities (EUROSTAT).

Turk, A. (1992), GIS Cogency: cognitive ergonomics in geographic information systems, PhD Thesis, Department of Surveying and Land Information, University of Melbourne.

Vandenberg, C., F. Tuijman, et al. (1997), *Multi-server Internet GIS: Standardization and Practical Experiments*, International Conference and Workshop on Interoperating Geographic Information Systems, 3-6 December 1997, Santa Barbara, CA, NCGIA, WWW (http://www.ncgia.ucsb.edu/conf/interop97/).

Vose, M. (1990), Hot Links to Go, Byte, **November:** pp. 373-377.

Wang, F. (1994), "Towards a natural language interface: an approach of fuzzy query," *International Journal of Geographical Information Systems* **8**(2): pp. 143-162.

Wang, F. (1994), "The use of artificial neural networks in a geographical information system for agricultural land-suitability assessment," *Environment and Planning A* **26**: pp. 265-284.

Wang, F., G. B. Hall, et al. (1990), "Fuzzy information representation and processing in conventional GIS software: database design and application," *International Journal of Geographical Information Systems* **4**(3): pp. 261-283.

Weber, A. (1929), *Theory of the location of industries*, Chicago, University of Chicago Press.

Wei, Z., H. Jianbang, et al. (1992), "Application of expert systems in land resources research," *Computers, Environment and Urban Systems* **16**: pp. 321-327.

Wharton, S. W. (1987), "A spectral-knowledge-based approach for urban land-cover discrimination," *IEEE Transactions on Geoscience and Remote Sensing* **GE-25**(3): pp. 272-282.

Willer, D. (1990), *A Spatial Decision Support System for Bank Location: A Case Study*, Technical Report 90-9, NCGIA, Santa Barbara, CA.

Worden, G. (1986), *Problems in defining high technology industries*, US Bureau of Census.

Yaar, M. (1994), Buckinghamshire County Council, *Personal communication*. 13 October 1994.

Yuan, M. (1996), *Temporal GIS and Spatio-Temporal Modelling*, Third International Conference/Workshop on Integrating GIS and Environmental Modelling, 21-25 January 1996, Santa Fe, New Mexico, NCGIA Publications Office, CD.

Zadeh, L. (1984), "Making computers think like people," *IEEE Spectrum* (August): pp. 26-32.

Appendix A

Genetic Algorithm Methodology

This appendix details the methodology adopted within GeoAnalyser for using a Genetic Algorithm to calibrate a spatial interaction model and how this compares with the general procedure for using GAs. The methodology is illustrated using the example of the demonstration application described in Chapter 5.

A.1 General Procedure

When using a GA, the following general procedure is used to evolve an optimal solution to a given problem (for an overview of the GA process see Chapter 2 Section 4.2):

1. Determine a representation for each individual in the population. Considerations include the number of chromosomes for each individual, the number of genes in each chromosome and the data type of the genes.

2. Create a fitness function. The fitness function returns a numeric value which rates an individual according to how successfully it solves the problem at hand.

3. Determine the configuration of the GA. The main variables are the number of pools and the number of individuals in each pool.

4. Determine the operators to be used (covering the stages of initialisation, selection, crossover, mutation and migration) and the associated rate parameters.

5. Create a prototype individual based on the chosen representation.

6. Seed the pool(s) with copies of the prototype individual using the chosen configuration and initialisation operators. Usually a set of individuals with random values for the chromosomes is created (i.e. using the random initialisation operator).

7. Determine the fitness of each individual using the fitness function.

8. Run the GA using the following subprocedure for the given number of generations or until a suitably optimal individual has been found:

i. Select individuals from the pool for crossover using the selection operator. Create new offspring using the crossover operator. Modify single individuals using the mutation operator. Migrate individuals between pools using the migration operator.

ii. Determine the fitness of the individuals.

iii. Discard individuals that perform poorly.

The representation of each individual is an important step in achieving a successful result with a GA. A poor representation can result in solutions being difficult to evolve or wasteful searches of the solution space. For GeoAnalyser, each individual must contain the set of model parameters to be optimised.

The QGAME library offers several data types (`integer`, `long` and `double`) for use in representing gene values. QGAME restricts genes of a chromosome to be of the same data type.

If a GA toolkit or library is used to assist in code building, then many of the above steps will be performed automatically by the library. In GeoAnalyser, most of the procedure described above is handled by the client using QGAME.

A.2 Methodology Used in GeoAnalyser Demonstration

In GeoAnalyser, each individual is represented in terms of a single chromosome containing one gene (of data type `double`) for each parameter in the spatial model. This ensures that all parameters are given equal weight in the evolution process. Thus for the model of high technology industry, each individual will contain a single chromosome with twelve genes. This representation is hard wired into the client at present. This could be a drawback if the modeller wishes to add or remove parameters (i.e. modelling factors) dynamically.

GeoAnalyser departs slightly from the procedure described above in its use of *ranges* for genes. This is a feature of QGAME whereby each gene can be constrained between a minimum and maximum value. This was used in the demonstration to limit the evolution of the model parameters within a window 30% above and below the value of the parameter in the prototype individual. Thus divergent solutions were largely avoided.

Configuration of the GA (covering the number of pools, pool sizes, genetic operators and rate parameters) is achieved using the various dialog boxes described in Chapter 4 Section 4. The QGAME class library offers the following set of operators:

Initialisation: random, initialise from file, combinatorial

Selection: truncated, roulette wheel, tournament, linear, stochastic remainder, window

Crossover: Holland, two point, uniform, combinatorial, chromosomal

Mutation: basic, combinatorial

Migration: best-to-worst

Facilities are also provided to add custom operators to the library quite easily.

For the demonstration application, various numbers of pools and pool sizes were used, using the following (default) operator and parameter set-up:

Initialisation: random

Selection: roulette wheel

Crossover: Holland (crossover rate = 0.6)

Mutation: basic (mutation rate = 0.03)

Migration: best-to-worst (migration rate = 0.3)

Note that the migration operator is only active where more than one pool of individuals is created, as it is used to migrate individuals between pools.

The spatial interaction model for high technology industry is loaded as a plug-in on the server. The fitness of an individual is thus calculated by sending an Apple Event (containing the model parameters of an individual and the time step to be used) from the client to the server. The server then runs the model and compares the results against the currently loaded data set on the server (in the demonstration, data for SE England between 1982 and 1986 was used) and the fitness calculated from the average error between the two sets of spatial data (see p93 for the actual error function used). The fitness value is then returned to the client.

After each generation, the permitted range of each parameter is re-calculated based on 30% above and below the spread of values of each parameter across the population. Thus effectively a moving window is created for a each parameter. As the individuals evolve, the window around each parameter should gradually widen, allowing exploration of different areas of the parameter space.

Appendix B

Model Calibration Results Using GeoAnalyser

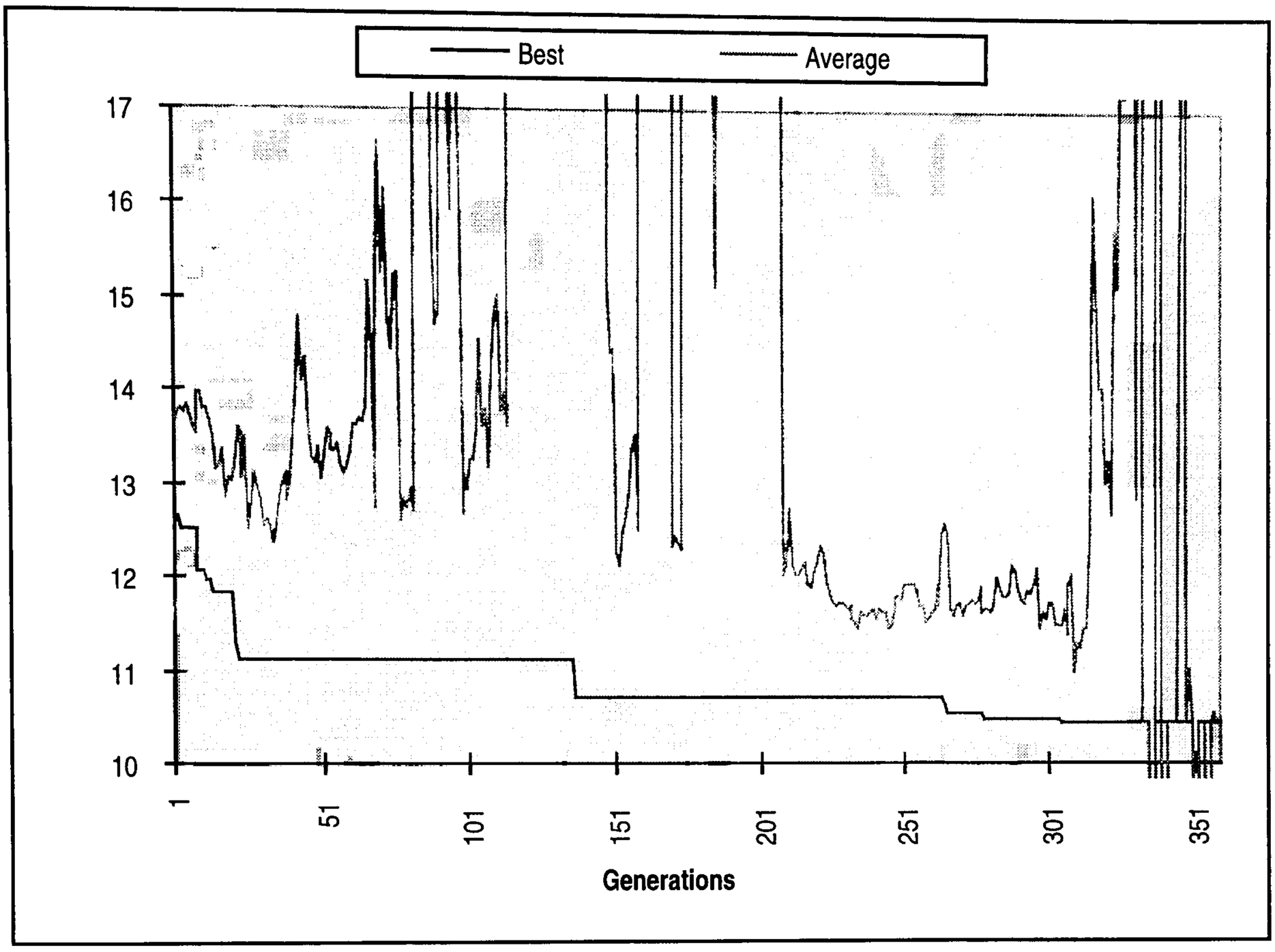


Figure B.1 – Calibration Results (best fitness = 10.43)

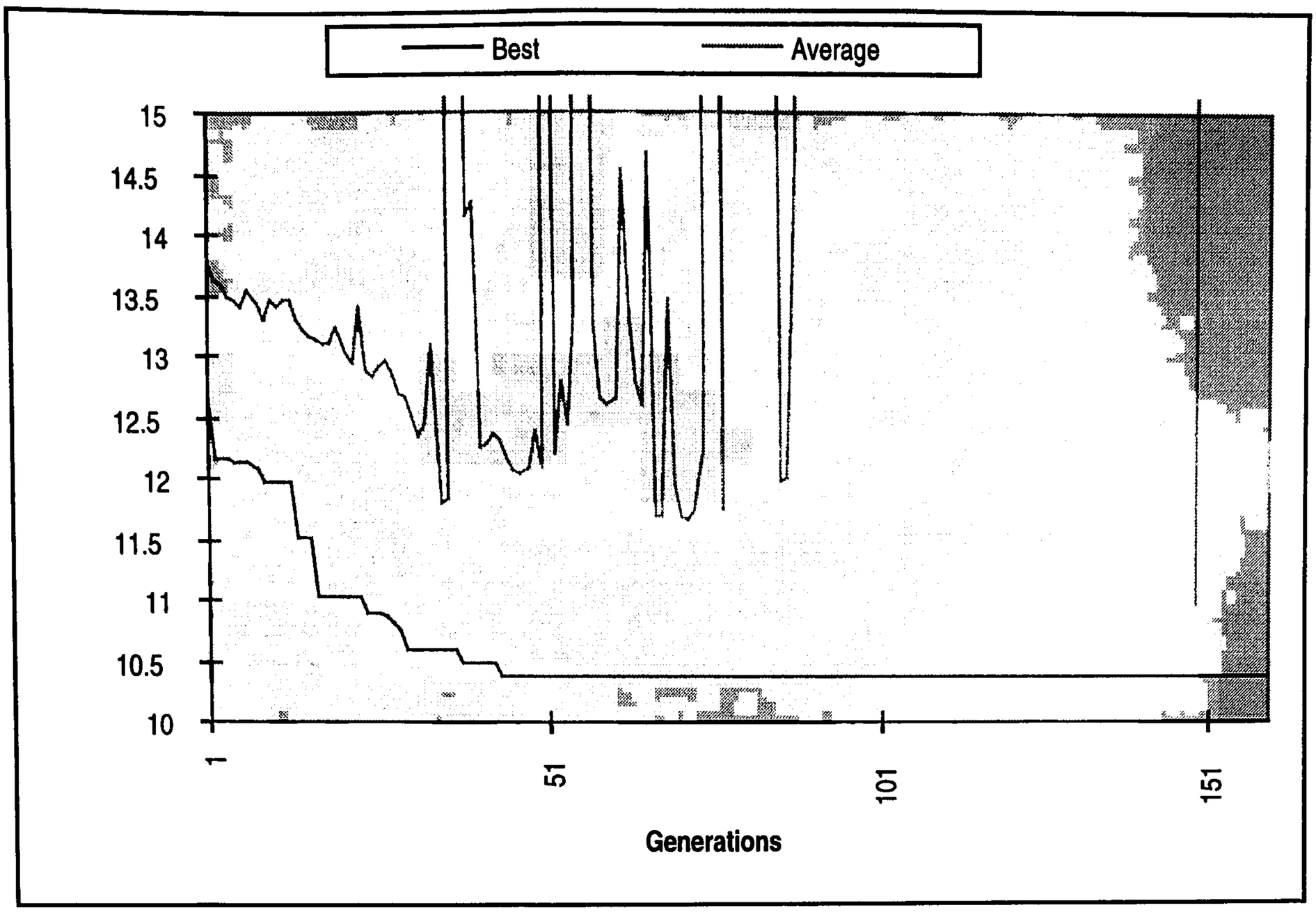


Figure B.2 – Calibration Results (best fitness = 10.39)

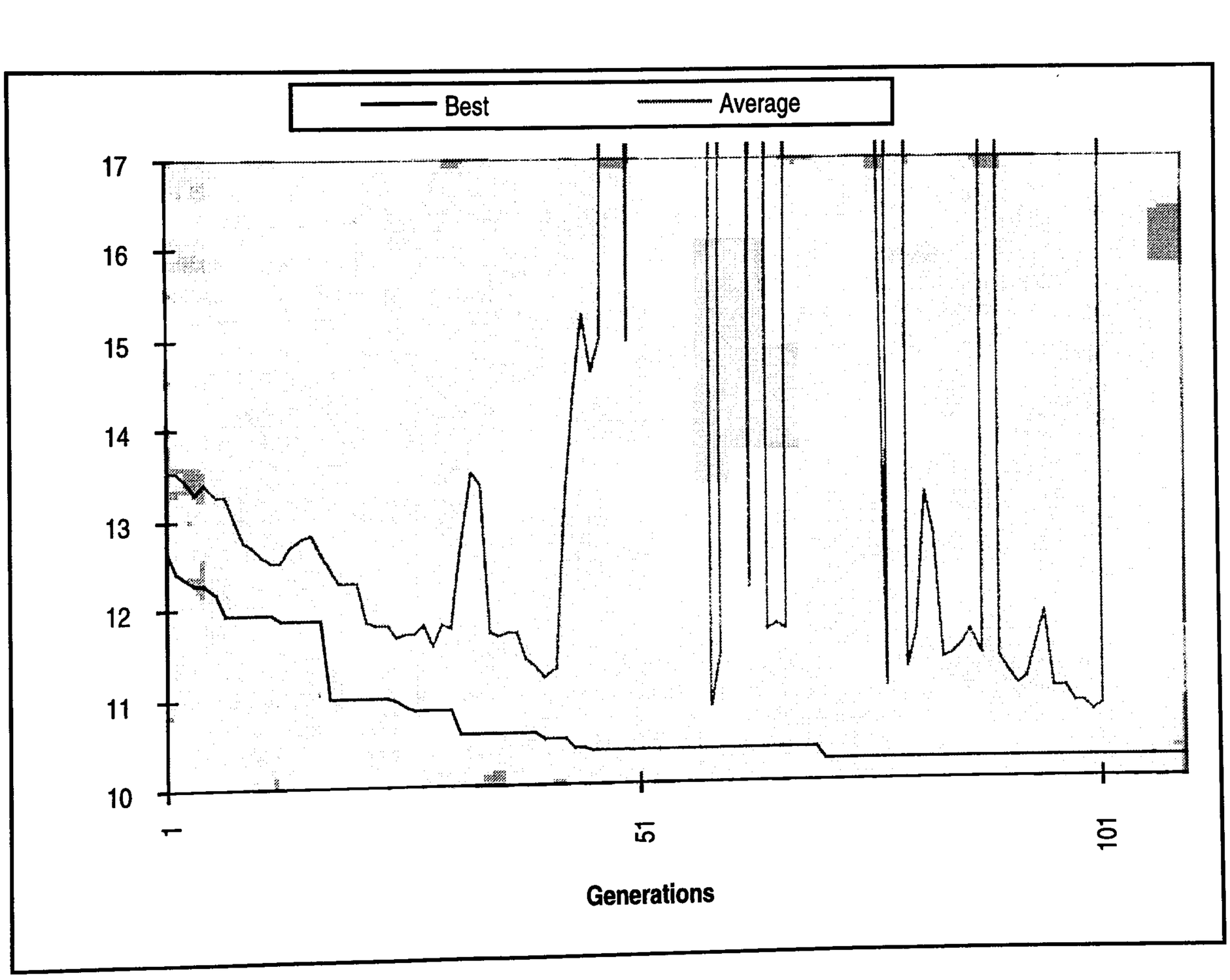


Figure B.3 – Calibration Results (best fitness = 10.25)

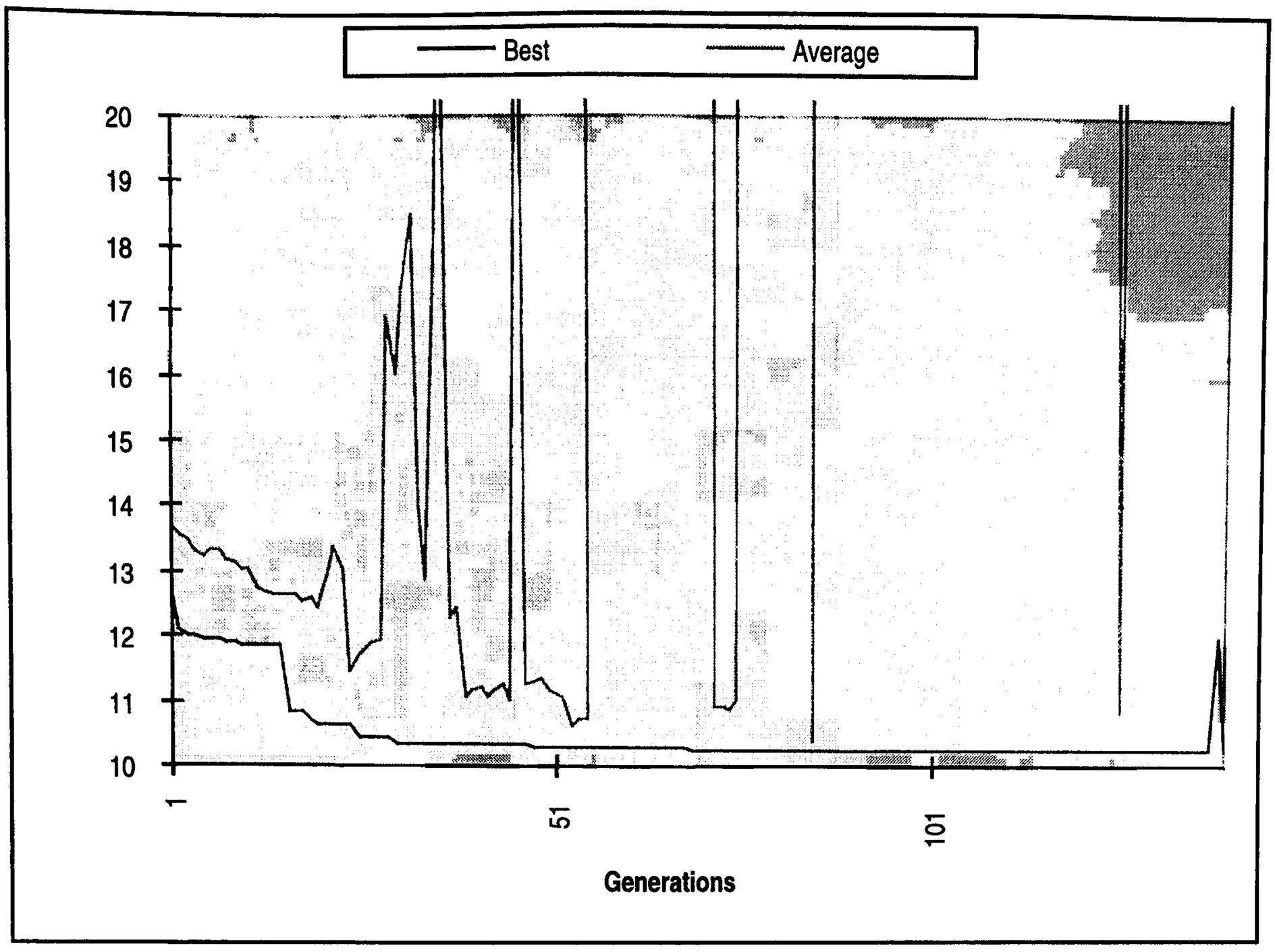


Figure B.4 – Calibration Results (best fitness = 10.24)

Glossary

API

Application Programmer's Interface, a set of software methods and procedures used typically to extend an existing software system.

Apple Event

The high level interapplication communication technology available on MacOS computers.

Chromosome

Sequence of genes used to represent an individual within a genetic algorithm.

Client-server model

Widespread model for building distributed applications, whereby operations on a client generally initiate communications to invoke further operations on a server application accessible on a network.

Co-operative coupling

Method of linking modelling/analysis methods with a spatial decision support system or GIS by employing interapplication communication methods.

Data-oriented systems

A class of spatial decision support system that employs intelligent systems techniques for data storage, query or conversion.

DDE

Dynamic Data Exchange – the low level message passing protocol used in the Microsoft Windows operating system.

Decision-oriented systems

A class of spatial decision support system that employs intelligent systems techniques for knowledge encapsulation and application in order to enhance decision support capabilities.

Design autonomy

The ability to design components of a system without being affected by local changes elsewhere in the system.

Direct co-operative coupling

Form of co-operative coupling where an analysis/modelling component communicates directly with a GIS.

Distributed systems

Computer systems that are able to co-operate by the exchange of messages over a network.

Event loop	The main event processing loop typically present in an application using a graphical user interface.
Expert system	An intelligent systems technique that represents knowledge in the form of rules in a knowledge base.
Fitness function	The function that is used to evaluate the fitness of an individual during the execution of a genetic algorithm.
Fuzzy logic	Intelligent systems technique that allows imprecise concepts to be modelled within rules (e.g. near, far, very etc.).
Gene	Basic unit information encoded within a chromosome.
Genetic algorithm	Intelligent systems technique for optimisation that is inspired by the process of natural evolution.
Genetic operator	Operator that is used to modify individuals in some way during the execution of a genetic algorithm. Examples include the mutation and crossover operators.
GIS	Geographical Information Systems.
GRE	Government Research Establishment.
HEI	Higher Education Institute.
Heterogeneous modular system	A modular system composed of both custom-built and off-the-shelf components.
Homogeneous modular system	A modular system composed of entirely custom-built components.
HTI	High technology industry.
IAC	Interapplication communication – the set of techniques that permit separate applications to communicate.
Ill-structured problems	Problems that are typically difficult to identify and rationalise and that cannot be solved using known procedures or methodologies. Analysis of such problems often reveals subproblems that are structured.

Implementation autonomy
The ability to implement components of a system without being affected by local changes elsewhere in the system.

Indirect co-operative coupling
Form of co-operative coupling where an intermediate component is present between an analysis/modelling component and the GIS.

ISDSS
Intelligent spatial decision support systems.

Loose coupling
Method of coupling models with GIS using files to exchange data.

Message passing
Software communication model where data exchanges are modelled as message objects sent between processes that are potentially distributed across a network.

Model-oriented systems
Class of spatial decision support system that employs intelligent systems techniques to model or analyse spatial data in order to generate understanding or new information as part of the process of generating new knowledge.

Neural network
Intelligent systems technique that is inspired by the structure of brain tissue and is particularly useful for pattern recognition.

Pipe
Popular method of synchronous communication between two processes on a UNIX system.

Rule induction
Intelligent systems technique that automatically builds rules (usually in the form of a decision tree) in order to model behaviour contained within a dataset.

SDSS
Spatial decision support systems.

Socket
Popular method of synchronous communication between two processes. Uses the TCP/IP message protocol.

Soft knowledge
Knowledge about a system that is difficult to express explicitly, but can be implicitly inferred, represented or modelled.

Spatial interaction model
Model of a spatial system where modelling entities are represented as points, lines or regions and their interactions are modelled individually using model equations expressed in terms of the modelling entities.

SQL	Structured Query Language, the industry standard language for querying relational databases.
TCP/IP	Transmission Control Protocol/Internet Protocol – the network communication protocol developed for workstations and popularised by the Internet.
Thematic map	Visualisation form common in geographical information systems that represents spatial data using map displays.
Tight coupling	Method of coupling models with GIS using APIs, hooks or other form of directly invoked procedures.
TIGER	Spatial data format developed by the US Census Bureau.